

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

A METHODOLOGY FOR THE DEVELOPMENT OF SECURE VERTICAL WEB PORTALS

by

Peter A. Wu

December 2002

Thesis Advisor:
Second Reader:

Neil C. Rowe
Xavier Maruyama

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

| | | | | |
|---|---|--|--|--|
| REPORT DOCUMENTATION PAGE | | | <i>Form Approved OMB No. 0704-0188</i> | |
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503. | | | | |
| 1. AGENCY USE ONLY (Leave blank) | | 2. REPORT DATE December 2002 | 3. REPORT TYPE AND DATES COVERED Master's Thesis | |
| 4. TITLE AND SUBTITLE: A Methodology for the Development of Secure Vertical Web Portals | | | 5. FUNDING NUMBERS | |
| 6. AUTHOR(S) Peter A. Wu | | | | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000 | | | 8. PERFORMING ORGANIZATION REPORT NUMBER | |
| 9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A | | | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER | |
| 11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. | | | | |
| 12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited | | | 12b. DISTRIBUTION CODE | |
| 13. ABSTRACT <p>In this thesis we investigate the development of vertical web portals (vortals) that fulfill targeted organizational mission needs. This specific type of portal provides narrow-scoped data, information and services while affording the user accessibility over a public network, such as the Internet. As part of the investigation, we present a methodology for architecting such portals with explicit consideration of security policy. The methodology, along with some preliminary guidelines, is intended to serve as a first approximation of a framework for both the development of vertical portals and the definition of doctrine on the application of vortals. We illustrate this methodology with an application to a Navy ship.</p> | | | | |
| 14. SUBJECT TERMS Vertical Web Portal, Secure Vertical Web Portal, Security Architecture, Security Policy, Knowledge Management, Portal, Vortal, Methodology | | | 15. NUMBER OF PAGES 127 | |
| | | | 16. PRICE CODE | |
| 17. SECURITY CLASSIFICATION OF REPORT Unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified | 20. LIMITATION OF ABSTRACT UL | |

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**A METHODOLOGY FOR THE DEVELOPMENT OF SECURE VERTICAL
WEB PORTALS**

Peter A. Wu
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1991

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

**NAVAL POSTGRADUATE SCHOOL
December 2002**

Author: Peter A. Wu

Approved by: Neil Rowe
Thesis Advisor

Xavier Maruyama
Second Reader

Peter Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In this thesis we investigate the development of vertical web portals (vortals) that fulfill targeted organizational mission needs. This specific type of portal provides narrow-scoped data, information and services while affording the user accessibility over a public network, such as the Internet. As part of the investigation, we present a methodology for architecting such portals with explicit consideration of security policy. The methodology, along with some preliminary guidelines, is intended to serve as a first approximation of a framework for both the development of vertical portals and the definition of doctrine on the application of vortals. We illustrate this methodology with an application to a Navy ship.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

| | | |
|----------------------------|---|-----------|
| I. | INTRODUCTION..... | 1 |
| A. | PROBLEM STATEMENT | 4 |
| II. | RELATED WORK | 7 |
| A. | AUTONOMY | 7 |
| B. | BRIO | 13 |
| C. | ORACLE | 15 |
| D. | PLUMTREE..... | 19 |
| E. | RELATED RESEARCH..... | 29 |
| III. | ANATOMY OF A VORTAL..... | 31 |
| A. | THE INFORMATION ENVIRONMENT | 31 |
| B. | ABSTRACT MODEL..... | 32 |
| C. | MODEL APPLICATION | 35 |
| D. | FUNCTIONAL COMPONENTS..... | 37 |
| 1. | Presentation Services [Presenter Layer]..... | 38 |
| 2. | Core Management System [Coordinator and Mediator Layers] .. | 40 |
| 3. | Low Level Communications System [Communicator Layer] | 41 |
| 4. | Content [Provider Layer] | 41 |
| 5. | Hardware | 41 |
| 6. | Additional Supporting Services | 42 |
| IV. | SECURE VORTAL DEVELOPMENT | 43 |
| A. | CONCEPT DEVELOPMENT..... | 43 |
| 1. | Higher Policy and Doctrine..... | 43 |
| 2. | Organizational Policies and Mission | 45 |
| 3. | User Input | 45 |
| B. | REQUIREMENTS DEVELOPMENT | 46 |
| 1. | Organizational Requirements..... | 47 |
| 2. | Organizational Security Policy | 47 |
| 3. | Vortal Goal Definition | 48 |
| 4. | Vortal Requirements | 50 |
| 5. | Vortal Security Policy..... | 52 |
| C. | ARCHITECTURE DEVELOPMENT..... | 54 |
| 1. | Organizational Information System (ORIS) Architecture | 54 |
| 2. | Vortal Architecture..... | 56 |
| 3. | Vortal Security Architecture | 57 |
| 4. | Design Assessment | 58 |
| D. | IMPLEMENTATION AND PROTOTYPING..... | 59 |
| 1. | Designing the Implementation Plan | 60 |
| 2. | Prototyping the Design | 61 |
| E. | TESTING, VALIDATION, EVALUATION, CERTIFICATION AND | |
| ACCREDITATION | 62 | |
| F. | FEEDBACK | 63 |

| | | |
|-------------|---|------------|
| V. | IMPLEMENTATION | 65 |
| A. | ASSEMBLE THE RIGHT TEAM..... | 66 |
| B. | CONCEPT & CONTENT DEVELOPMENT | 67 |
| | 1. Organizational Policies and Mission | 67 |
| | 2. User Inputs | 67 |
| | 3. Vortal Target Audience..... | 68 |
| | 4. Vortal Content..... | 68 |
| C. | REQUIREMENTS DEVELOPMENT | 69 |
| | 1. Organizational Requirements..... | 69 |
| | 2. Organizational Security Policy | 70 |
| | 3. Vortal Goal Definition | 71 |
| | 4. Vortal Requirements | 71 |
| | a. Content Requirements..... | 72 |
| | b. Content Correlation | 78 |
| | c. Interface Requirements..... | 80 |
| | d. Hardware Requirements..... | 80 |
| | e. Software Selection..... | 82 |
| | 5. Vortal Security Policy..... | 83 |
| D. | ARCHITECTURE | 85 |
| | 1. Architecture..... | 86 |
| | 2. Design Assessment | 89 |
| E. | PROTOTYPING..... | 89 |
| F. | DEVELOP TRAINING, MAINTENANCE & ADMINISTRATION PLANS | 89 |
| VI. | COMPUTER SUPPORT FOR DOCTRINE AND POLICY | 93 |
| VII. | CONCLUSIONS | 95 |
| A. | FUTURE WORK..... | 95 |
| | APPENDIX A. LIST OF ACRONYMS..... | 97 |
| | APPENDIX B. HUMAN-COMPUTER INTERACTION AND USABILITY RESOURCES | 101 |
| | APPENDIX C. DECISION MAKING RESOURCES | 103 |
| | LIST OF REFERENCES..... | 105 |
| | INITIAL DISTRIBUTION LIST | 111 |

LIST OF FIGURES

| | | |
|------------|---|----|
| Figure 1. | Portal Architecture (From White, 2002)..... | 1 |
| Figure 2. | Knowledge Sources (From Reynolds & Koulopoulos, 1999) | 3 |
| Figure 3. | Autonomy Portal Architecture (From Autonomy, 2002) | 8 |
| Figure 4. | Autonomy Logical Architecture (From Autonomy, 2002)..... | 9 |
| Figure 5. | Autonomy Security Architecture (From Autonomy, 2002a)..... | 10 |
| Figure 6. | Non-Mapped vs. Mapped Security Mechanisms (From Autonomy, 2002a)... | 12 |
| Figure 7. | Job Factories and Agents manage work (From Brio, 2001) | 13 |
| Figure 8. | Portal Architecture-Oracle <i>9iAS</i> (From Oracle, 2002a) | 15 |
| Figure 9. | Plumtree Portal Architecture (From Plumtree, 2002)..... | 19 |
| Figure 10. | Plumtree Password-Based Authentication (From Plumtree, 2000) | 22 |
| Figure 11. | Authentication by External Directory Service (Plumtree, 2000)..... | 23 |
| Figure 12. | Four Access Control Check Points (From Plumtree, 2000). | 26 |
| Figure 13. | Five Layer MIA (From Gardner, 1999)..... | 32 |
| Figure 14. | Presenter Layer (From Gardner, 1999)..... | 33 |
| Figure 15. | Coordinator Layer (From Gardner, 1999) | 34 |
| Figure 16. | Mediator Layer (From Gardner, 1999)..... | 34 |
| Figure 17. | Communicator Layer (From Gardner, 1999)..... | 35 |
| Figure 18. | Simple Vortal Architecture (Adapted From Powell, 2000) | 36 |
| Figure 19. | Basic Portal Architecture (From White, 2001)..... | 38 |
| Figure 20. | HCI Development Guidance (From DoD, 2001)..... | 39 |
| Figure 21. | Human-Computer Interaction Process (From Hewett, 1992)..... | 40 |
| Figure 22. | Secure Vortal Architecture Development Process Diagram..... | 44 |
| Figure 23. | Component Relationships within Vortal Requirements and Security Policy Development | 46 |
| Figure 24. | Vortal Goal Components | 48 |
| Figure 25. | High Level ORIS Architectural Environment | 54 |
| Figure 26. | High Level ORIS Architecture Refined..... | 55 |
| Figure 27. | Example ORIS Security Architecture—Dial Up Network | 56 |
| Figure 28. | “Basic” Discover-Locate-Request-Deliver Use Case (From Gardner, 1999a) | 72 |
| Figure 29. | “Detail” Use Case Describing the Discovery Level of Resources (From Gardner, 1999a) | 73 |
| Figure 30. | “Enter” Use Case for System-Entry Behavior (From Gardner, 1999a)..... | 74 |
| Figure 31. | “Exit” Use Case Describing System Exit Behaviors (From Gardner, 1999a) | 75 |
| Figure 32. | “Discover” Use Case Describing Resource Discovery Behaviors (From Gardner, 1999a) | 76 |
| Figure 33. | Web Service Provides Interoperability Across Application Servers (From Plumtree, 2002b)..... | 82 |
| Figure 34. | USS NEVERSAIL Logical Vortal Architecture (From Plumtree, 2002b)..... | 85 |

| | | |
|------------|--|----|
| Figure 35. | Proposed USS NEVERSAIL Vortal Architecture (Adapted From Plumtree, 2002c)..... | 86 |
| Figure 36. | Proposed USS NEVERSAIL Vortal Security Architecture..... | 87 |

LIST OF TABLES

| | | |
|----------|--|----|
| Table 1 | Portal Type Classifications | 4 |
| Table 2 | Default User Roles and Privileges to Objects (From Plumtree, 2000) | 25 |
| Table 3 | Crawlers Used To Import Increasingly Sensitive Information From A Host System. (From Plumtree, 2000) | 28 |
| Table 4 | Correlation Table For Combined Resource and Target Audience to Requirements | 69 |
| Table 5 | “Enter” Use Case Scenario Under Normal Conditions and With Several Variations (Adapted From Gardner, 1999a) | 75 |
| Table 6 | “Discover” Use Case Scenario Under Normal Conditions and With Several Variations (Adapted From Gardner, 1999a) | 76 |
| Table 7 | Default User Roles and Privileges to Objects (Adapted From Plumtree, 2000) | 78 |
| Table 8 | Assigned User Roles to Resources | 79 |
| Table 9 | Prioritized USS NEVERSAIL Vortal Resources With Associated Availability Requirements. | 81 |
| Table 10 | Correlation of User Content Levels to Resources..... | 83 |

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Throughout these last sixteen years that I have been in uniform, I have truly learned the value of education and training. I would like to thank the U.S. Navy for the privilege of allowing me to join such a world-class outfit as well as for providing me the opportunities to earn both baccalaureate and graduate degrees. I would like to thank Dr. Neil Rowe and Dr. Xavier Maruyama for overseeing my research study of this topic and for their courage and kindness in the face of adversity. Lastly, I would especially like to thank my family for their undying devotion and support throughout the span of my career and studies. *Non Sibi Sed Patriae.* Go Navy!

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A web portal or portal is a “web site or service that offers a broad array of resources and services such as e-mail, forums, search engines, and on-line shopping (Webopedia.com, 2002).” In World Wide Web information management, the portal has become king. What defines the portal in this role comes down to its flexibility in allowing collaboration, integration, aggregation and consolidation of data, information and services into a single user interface that is accessible through any standard Internet web browser.

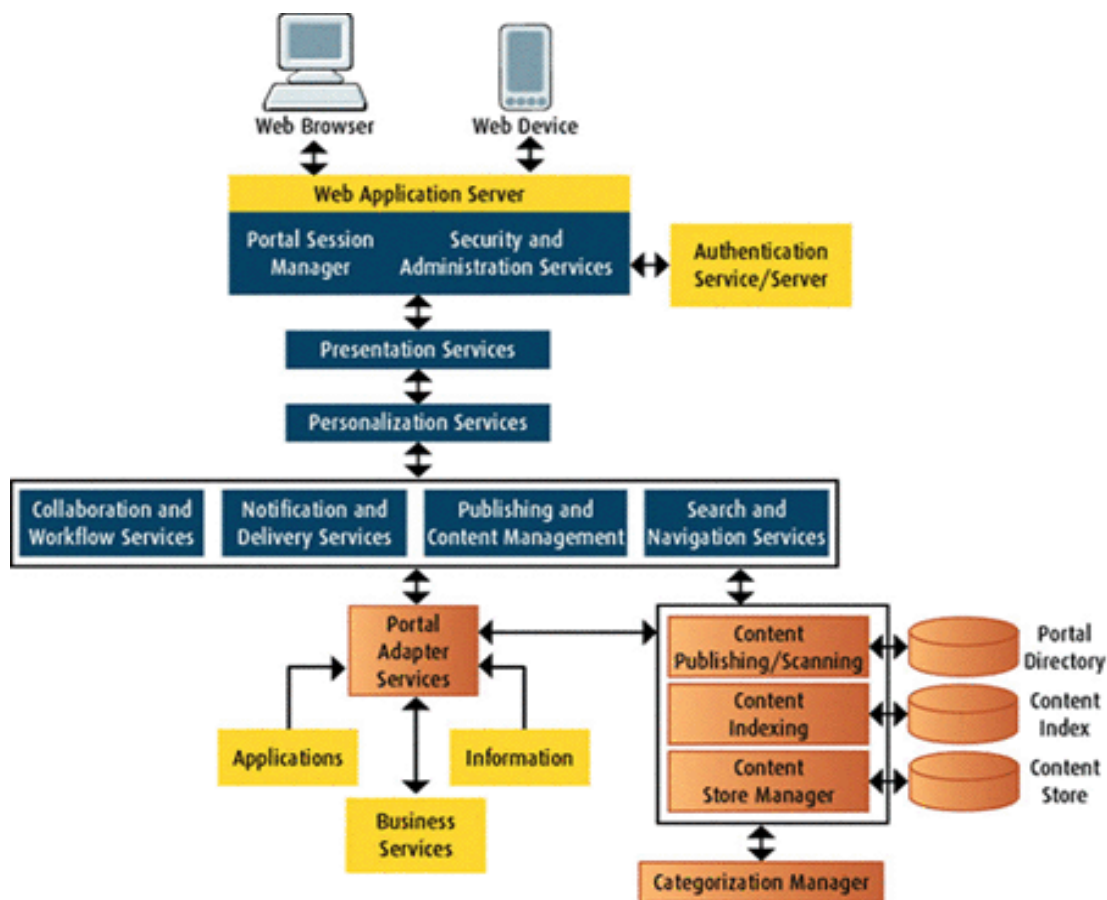


Figure 1. Portal Architecture (From White, 2002)

Portals are very powerful tools which provide a means to deal with information overload in the digital world (Dias, 2001). They can be harnessed in various ways to increase the productivity of workers as well as increase delivery of managed information

and services to the public. For modern enterprises, portals can also be used to provide a single gateway to all enterprise information and knowledge resources (Collins, 1999). Figure 1 illustrates the main components of a typical portal and how content flows from network systems to the end user.

In her paper on corporate portals, Dias classifies portals in two ways: environment and function. Portals classified according to environment are either public or corporate in nature. Public portals provide a single interface to the immense network of servers that is the Internet. These portals, sometimes known as consumer portals, establish a one-way relationship with their patrons and are usually exploited as a new marketing media. For example, they may be used in a manner similar to television, radio or the press for the purpose of marketing. On the other hand, corporate portals provide business-specific information in a context that helps users find information needed to face their competitors. This flavor of portal seeks to provide an integrated environment that supports information access, delivery, and work-support across organizational dimensions. Figure 2 illustrates the typical knowledge sources for integration within an organization. (Reynolds & Koulopoulos, 1999)

Portals that are classified according to their function are either collaborative processing and/or decision-supportive in nature (Dias, 2001). Collaborative-processing portals use some sort of collaborative communications tool to improve productivity from manipulation of information developed by corporate applications, the traditional supply chain as well as individuals and groups. These types of portals go beyond mere “content retrieval” to become a powerful device for discussing and sharing business content with other users (White, 1999). In contrast, decision-supportive portals assist managers, executives, and analysts to obtain required corporate information for the purpose of developing consistent business decisions (White, 1999).

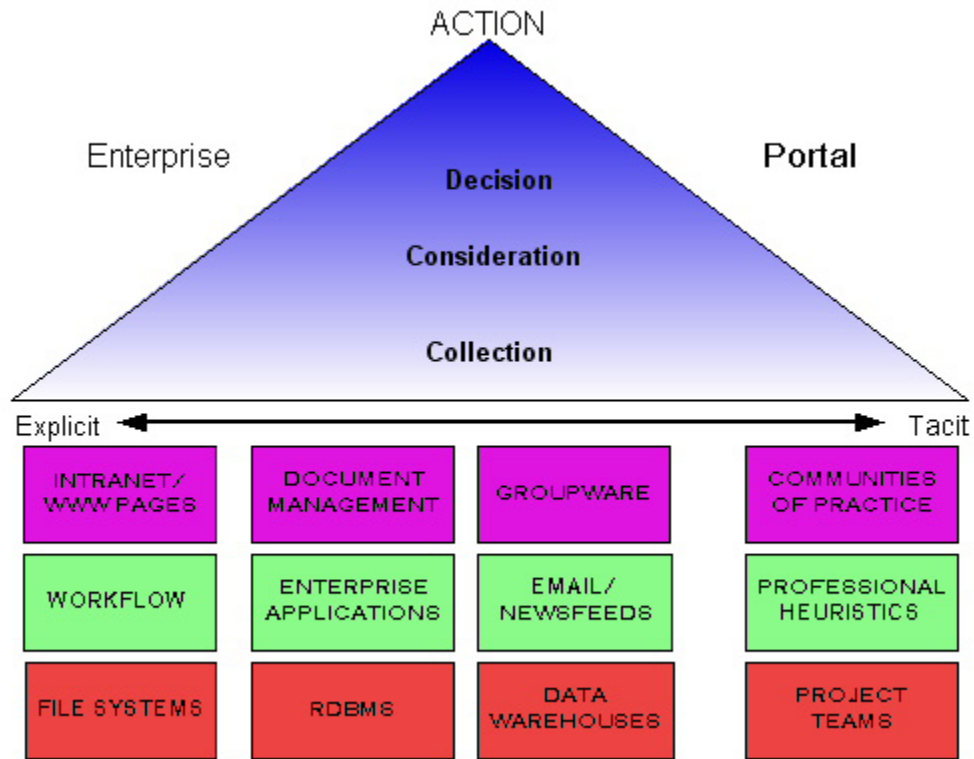


Figure 2. Knowledge Sources (From Reynolds & Koulopoulos, 1999)

Portals can also be classified according to industry scope. These types of portals are categorized in two ways: Vertical and Horizontal. A vertical industry is defined as an industry with a relatively narrow range of information and services. A vortal (vertical industry portal) is “a web site that provides a gateway or portal to information related to a particular industry such as health care, insurance, automobiles, or food manufacturing (SearchEBusiness.com, 2002).”

Vertical industry portals provide specific narrow-scoped data, information, and services through a single user interface. For example, a web portal dedicated to the real-estate industry with supporting data, information and services related to real property could be considered a vortal. Vertical industry portals also encompass enterprise information portals. In contrast, a horizontal industry is focused on a wide range of information and services. Horizontal portals provide a wide range of information and services in one convenient web interface. These portals are familiar to most people from search engines like Yahoo!, Excite and Alta Vista. Since many industries tend to

specialize, they are more often vertical in nature. Table 1 illustrates these portal type classifications.

| CLASSIFICATION | | <i>Environment</i> | <i>Function</i> | <i>Scope</i> |
|----------------|---|--------------------|-----------------|--------------|
| TYPE | <i>Public</i> | ✓ | | |
| | <i>Corporate</i> | ✓ | | |
| | <i>Collaborative Processing</i> | | ✓ | |
| | <i>Decision-Supportive</i> | | ✓ | |
| | <i>Collaborative Processing & Decision Supportive</i> | | ✓ | |
| | <i>Vertical Industry (VORTAL)</i> | | | ✓ |
| | <i>Horizontal Industry</i> | | | ✓ |

Table 1 Portal Type Classifications

A. PROBLEM STATEMENT

In this thesis we take the concept of a vertical industry web portal coupled with well-established principles of software design and suggest a methodology for the development of a secure vortal. Specifically, we propose a methodology and a security architecture that can be used as a framework for the development and deployment of a secure vortal.

Many organizations struggle to find a structured approach to develop and implement information assurance within their enterprise information systems. Some researchers believe that effective authentication and intruder detection processes coupled with an effective strategy for handling intrusion dynamics can provide adequate information assurance (Satti & Garner, 2001). Other researchers advocate using deception as a means of information protection and network defense (Cohen, 1998). In the end, most researchers would agree that information assurance requires a variety of methods.

Thus, effectively planned information assurance within information systems requires a structured approach to security architecture development and implementation. This methodology should:

- Show applicability across a diverse set of information system architectures.
- Provide a methodology for security-requirement analysis, security-policy development, and a mechanism to determine where security services are provided within the architecture.
- Be straightforward enough to be understood by customers.
- Effectively and efficiently facilitate development of integrated security architectures for enterprise networks.
- Facilitate an integrated solution across complex heterogeneous information systems.
- Provide a final product that covers all guidelines, policies and customer requirements.
- Provide the customer with visibility into the process and developed solution.
- Facilitate incorporation of technology upgrades as well as policy and guideline revisions. (Lowman & Mosier, 1997)

Structured methodology for the development of information-system architectures should include security architecture. When a reproducible methodology is used in concert with specific standards such as the Joint Technical Architecture, then the required guidance is available to develop the necessary security architecture (DoD JTA, 2001). In addition, the implementation of one or more security policies through the use of a defined security architecture coupled with a variety of security services provides a reasonable layered approach to implementing configurable information assurance within a computer network.

A public network such as the Internet provides a unique environment to demonstrate the balance between information assurance and information accessibility. Defining a methodology for the development of a vortal possessing adequate information assurance provides a framework for future deployment of similar mechanisms that provide services to users on a public network like the Internet.

The primary research question addressed in this thesis is “by what methodology should information assurance principles and techniques be applied to support the development of a secure vertical web portal?” In addition, the following questions will be considered:

- How is it possible to prioritize the information assurance mechanism(s) of a vertical portal in order to obtain the most optimum secure system architecture?
- Within the context of information assurance, what method(s) should be applied in order to determine which security services would provide a reasonable level of information assurance while permitting a high level of information accessibility?

The scope of this thesis includes a general review of portals with an emphasis on providing a methodology for the development of architecture for a secure vortal for the Department of the Navy. Doctrine and policy with respect to computer support will be addressed but will not be developed within this thesis. But vortal implementation and testing are beyond the scope of this thesis and may provide subject matter for follow-on thesis work.

Currently the United States Navy has an organizational entity called Task Force Web whose mission is “to provide integrated and transformational information exchange for both the ashore and afloat Navy to take full advantage of Navy's IT21 and Navy and Marine Corps Intranet infrastructure investments” (Task Force Web, 2000). Task Force Web has the responsibility to oversee the development of the Navy Portal. This thesis can contribute to its mission and benefit all members of the U. S. Navy who use the greater Navy Portal and similar portals.

II. RELATED WORK

To fulfill the needs of industry and explore a profit area with considerable potential, numerous software companies have developed ready-made portal products that can be implemented with the addition of content and some minor configuration changes. These products are tailor-able to the needs of the customer organization. The following examples illustrate what some companies have fielded as their attempt at a portal “solution”¹.

A. AUTONOMY

Autonomy’s Portal-in-a-Box™ (Autonomy, 2000a) supports automated user profiling as well as allows users to use content from various sources through automated content aggregation and management on platforms such as Microsoft Windows NT, Microsoft Windows 2000, SUN Solaris, LINUX, HP-UX, Java Servlet Engine/Application Server, and Standard Web Servers. This uses Autonomy’s Dynamic Reasoning Engine (DRE™), a “unique pattern-recognition technology that automatically analyzes information based on its content.” Through statistically predictable word patterns, Autonomy’s DRE™ technology independently represents concepts and functions in more than 20 languages. This allows for automation of “the most critical processes including categorization, personalization, hypertext-link management, and highly personalized information delivery.” Figure 3 illustrates the Autonomy Portal architecture.

¹ The selection of these examples was based on the availability of architectural and security information and does not constitute endorsement for these products.

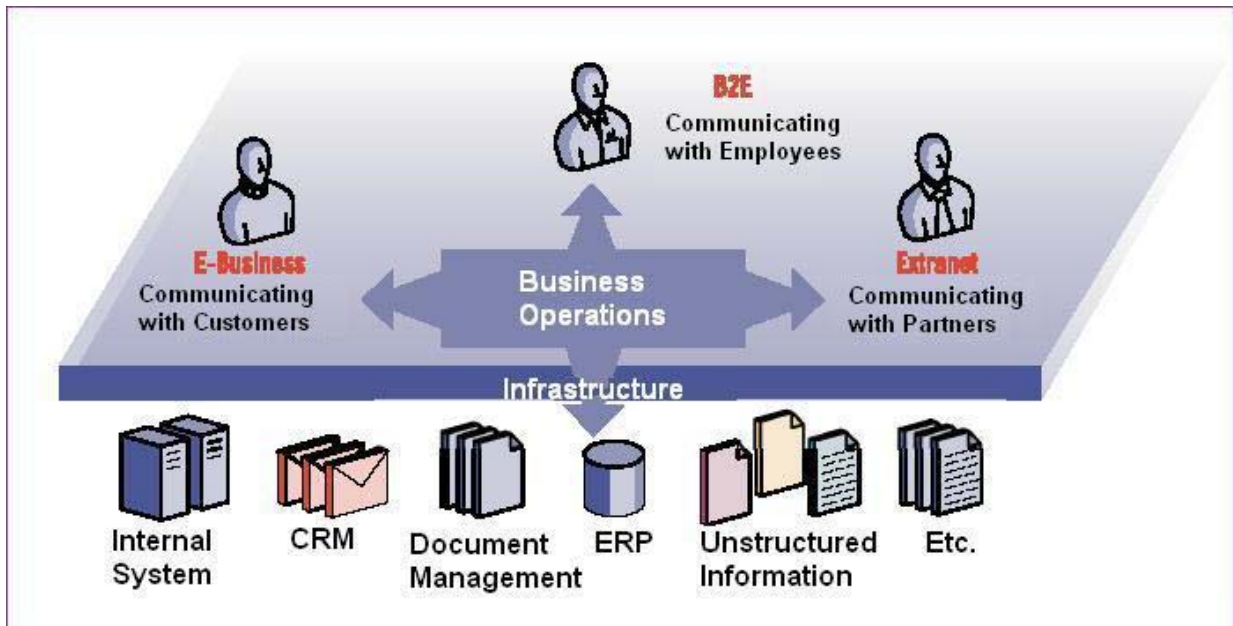


Figure 3. Autonomy Portal Architecture (From Autonomy, 2002)

An Autonomy portal “automatically processes digital content and allows enterprise applications to communicate with each other” through the use of a plug-in technology called Autonomy Content Infrastructure™ (ACI™). The ACI™ supports open Internet standards like “HTML, SGML, Text files, XML, binary delimited” and provides options to support over 200 data formats, audio, and repositories such as “Documentum, FileNET, Lotus Notes, Microsoft Exchange, NNTP servers, ODBC, Oracle, and POP3, among others.” Figure 4 displays the Autonomy’s so-called “logical architecture”.

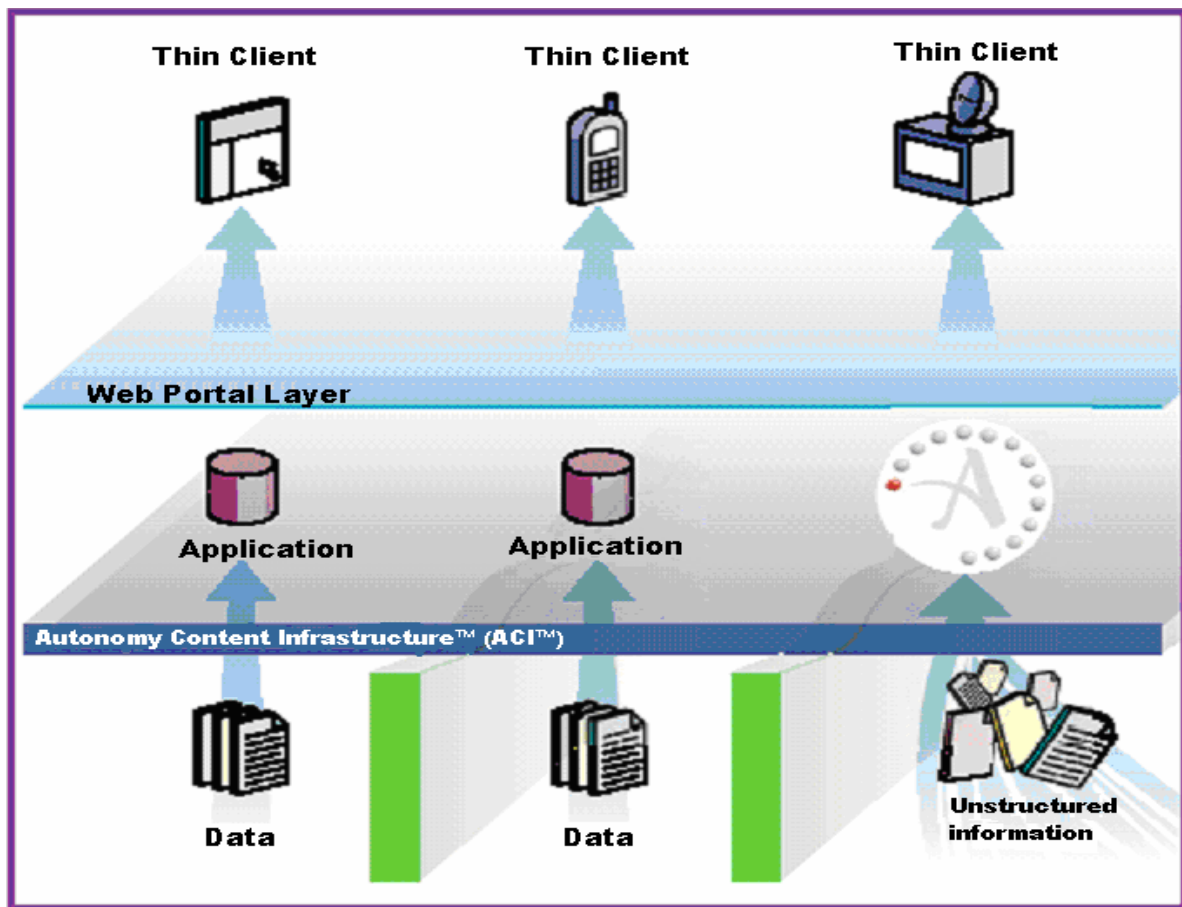


Figure 4. Autonomy Logical Architecture (From Autonomy, 2002)

“Typical security implementations require a combination of secure configurations of the front-end (ASPs, CGIs, Desktop applications, etc) and the back-end processes, namely the fetches and the DRE™.” Depending on the security granularity required for the system, a typical deployment of Autonomy’s Portal-in-a-Box™ uses a composite security architecture which includes secure access, user authentication, user entitlement, secure communications, and external secure site access. Figure 5 displays Autonomy’s security architecture.

Secure access is achieved by capitalizing on the inherent organizational network information system’s security architecture provided by firewalls and other network device configurations.

User authentication is achieved through the use of a combination of Autonomy’s own internal authentication and other authentication mechanisms such as Windows 2000

authentication, Light-weight Directory Access Protocol (LDAP), other standardized application authentication capabilities provided in products such as Lotus Notes or Microsoft Exchange as well as available non-standardized authentication mechanisms via Autonomy's application programming interface (API) tools.

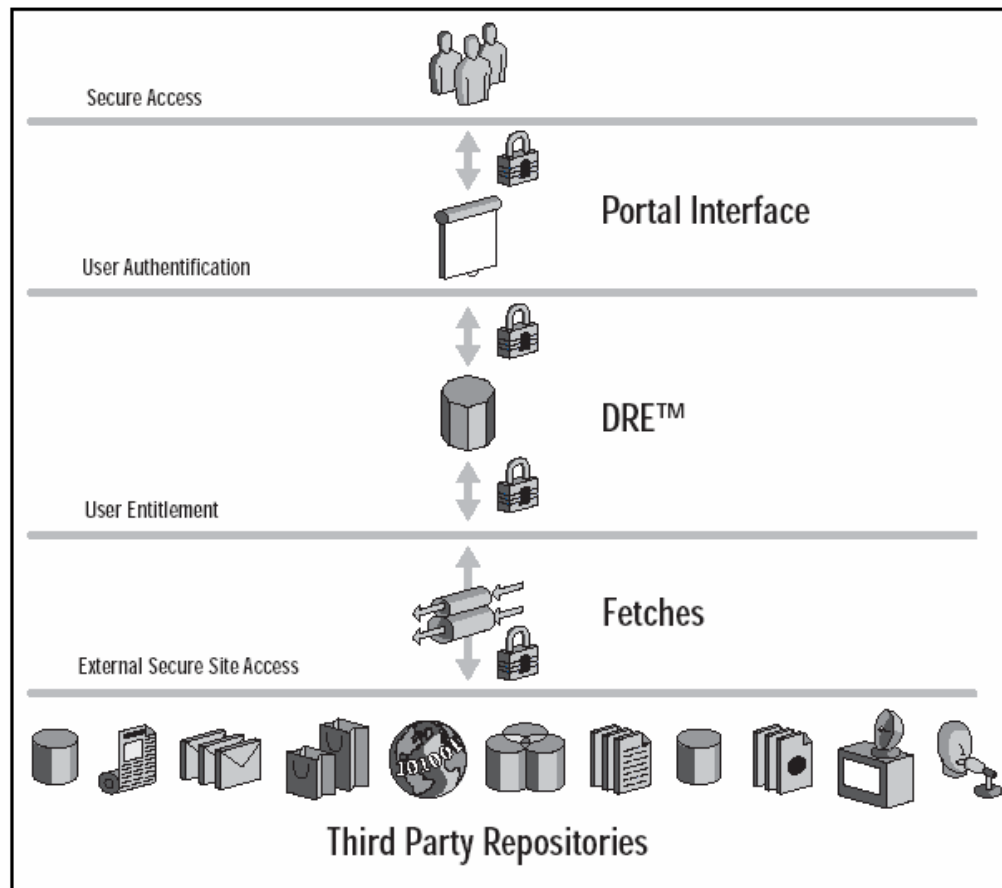


Figure 5. Autonomy Security Architecture (From Autonomy, 2002a)

User entitlements are controlled through the use of “proprietary entitlement strategies usually in the form of Access Control Lists (ACLs)” which are present in a wide range of content sources and formats. These entitlement strategies include third-party repository authorization, database segmentation via user roles, and entitlements based on field restrictions. (Autonomy, 2002a)

Third-party repository authorizations are content-repository implemented mechanisms. They add flexibility because content access is determined by “operating system authentication mechanisms or specific authorization CGIs, ASPs, etc.” at the time

the user attempts to view the original document. Their advantage is implementation simplicity; their disadvantage is that a user could potentially be shown results they are not entitled to view.

Database segmentation may be done via user roles. This means that DRE™ databases can perform role restriction. “Users are assigned to one or more roles and entitlement information is set for each one by identifying which databases each role is allowed to access.” Since DRE™ databases are information folders that divide the knowledge base in a logical manner; the DRE™ receives queries only pertaining to the applicable databases to which the user associated role is allowed to view. Their advantage is efficiency in that “unnecessary security processing is removed from the DRE™ and is done at the front-end level.” Their disadvantages are that entitlement is enforced at the application front end and anyone with direct access to the DRE™ back-end could “potentially have access to unauthorized databases.”

Entitlements based on field restrictions can be used as well. This means that entitlements are determined from restrictions placed on entitlement fields and filtering at the document level. Each document is assigned an entitlement field value at indexing time, and this at query time gets compared in some manner with the user’s entitlement value to determine access entitlement. Third-Party Entitlement Implementation (DRE™ plug-ins) may be used to restrict user entitlement. This means “DRE™ security plug-in modules ensure that only documents which the user is entitled to see are sent back to the requesting application.” This is implemented through either non-mapped or mapped security modes.

For non-mapped security mode, the DRE™ needs direct connectivity to the data repository to use any necessary APIs or plug-ins to obtain access control information. Plug-ins are available in the form of shared objects (for UNIX) and Dynamic Link Libraries (for Windows NT) “which can be custom built and easily plugged into the system.” For mapped security mode, “the Access Control List (ACL) of the documents in the repository is mapped into a structured field in the DRE™.” To ensure confidentiality, structured fields are encrypted. At the time of indexing, the indexing module retrieves each document ACL and stores it in the DRE™. The DRE™ does not

require any repository connection. Figure 6 compares Mapped security mechanisms with Non-Mapped security mechanisms.

| | NON MAPPED | MAPPED |
|------------------|---|---|
| ADVANTAGES | 100% in sync with the entitlement information in the repository. | Much faster response time. From an infrastructure point of view, the DRE™ does not need direct connection with the content repository. |
| DRAWBACKS | Response can be slow as entitlement lookup is done for each result sent back by the DRE™. Entitlement lookup is done by connecting to the original repository, which increases the overhead. Caching can be used to alleviate this. Also the DRE™ needs to process as many documents as necessary to find relevant results to send back. The worst case scenario would be if the user is entitled to see none of the documents, the DRE™ would have to process every single relevant document in the database to finally say that there are no results. | There will be a slight lag between a security setting being changed in the content repository and it being indexed into the DRE™, and hence available to the application. |
| USAGE | Good for environments where the security entitlement for documents changes frequently and where the average user is, on average, allowed to see more than 80% of the data. | Good for environments where the security entitlement does not change too often. |
| REQUIRED MODULES | Non mapped version of the security plug in DLLs for the repositories used. | Mapped version of the security plug-in DLL and a version of the indexing module that retrieves ACLs. |

Figure 6. Non-Mapped vs. Mapped Security Mechanisms (From Autonomy, 2002a)

Secure communication is achieved through Secure Socket Layer (SSL) encryption over the hypertext transfer protocol (HTTP). External Secure Site Access is achieved through the use of Autonomy's HTTP fetch and can exploit a variety of login and authorization methods such as HTTP authentication (username and password), form-based login using the POST method (HTTP fetch emulation of an HTML authentication

form), form-based login using the GET method (use of a specified branded Universal Resource Locator (URL) for authentication process activation), cookie-based login (using cookies to circumvent the login process), NT Login Management (NTLM) security (use of Windows NT login security provided at a website), and Client side SSL certificates (use of a client side SSL certificate authentication when fetching from a secure site).

B. BRIO

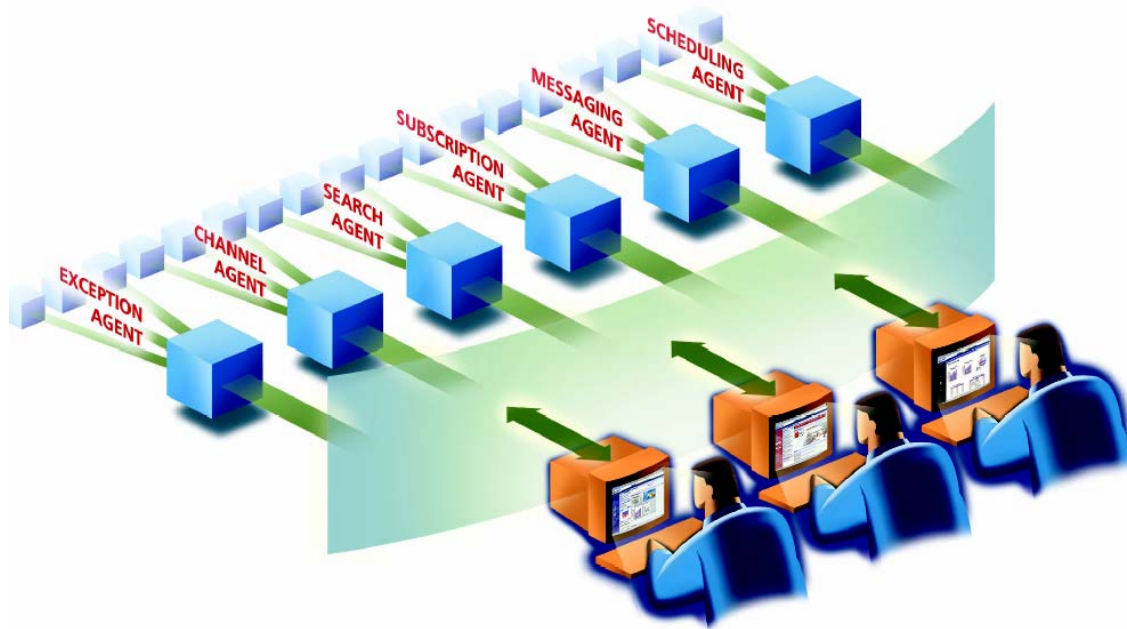


Figure 7. Job Factories and Agents manage work (From Brio, 2001)

The Brio Portal™ implementation is a bit different (Brio, 2001). Using a Java-based multi-tier architecture; the Brio Portal™ allows real-time performance evaluation with personalized metrics, content, and the ability to access both unstructured and structured data. Figure 7 suggests how personalized content “agents” automate information discovery and delivery from multiple data sources such as data warehouses, web sites and internal file systems. This affords the opportunity to leverage “applications, production and warehouse data as well as unstructured information including documents and web content, facilitating simplified, insightful decision-making.” To support this design, a series of component controls are used to handle the various tasks and services internal to the portal.

Using Autonomy's pattern-matching technology allows the Brio Knowledge Server component to search for "content in any language and format, from any location, and present it automatically with hyperlinks to similar information" through an active client service component WebClient. The WebClient component controls what users see and access through the portal. A Name Server component is used to manage service-agent configuration information, authentication, and initialization service as well as provide directory lookup. Session management is provided through a Service Broker that acts as a service-agent gateway server and dynamic load balancer to handle user requests. An Authentication Service component provides WebClient with client-service authentication through an internal authentication driver or an external authentication service.

The Repository manages object storage, search, browsing, or retrieval actions on the object repository. The Job Factory "requests services from external DBMS or Enterprise Resource Planning (ERP) systems and delivers the job output both to the end-user, through the WebClient, and to the Repository, for future distribution." An Event Server schedules Job Factory agents as well as provides users with a subscribable notification service. A centralized Administrator manages objects, categories, users, schedules, and services. An Integrator uses a Java Application Program Interface to integrate other systems with the Brio Portal. Security is achieved through profile-driven content and object-level access that "integrates with other user-authentication mechanisms including NT domains, LDAP, NIS/NIS+ and third party single sign-on security." Finally, the use of the Wireless Markup Language (WML) supports two-way interaction for wireless devices through wireless Internet service providers.

C. ORACLE

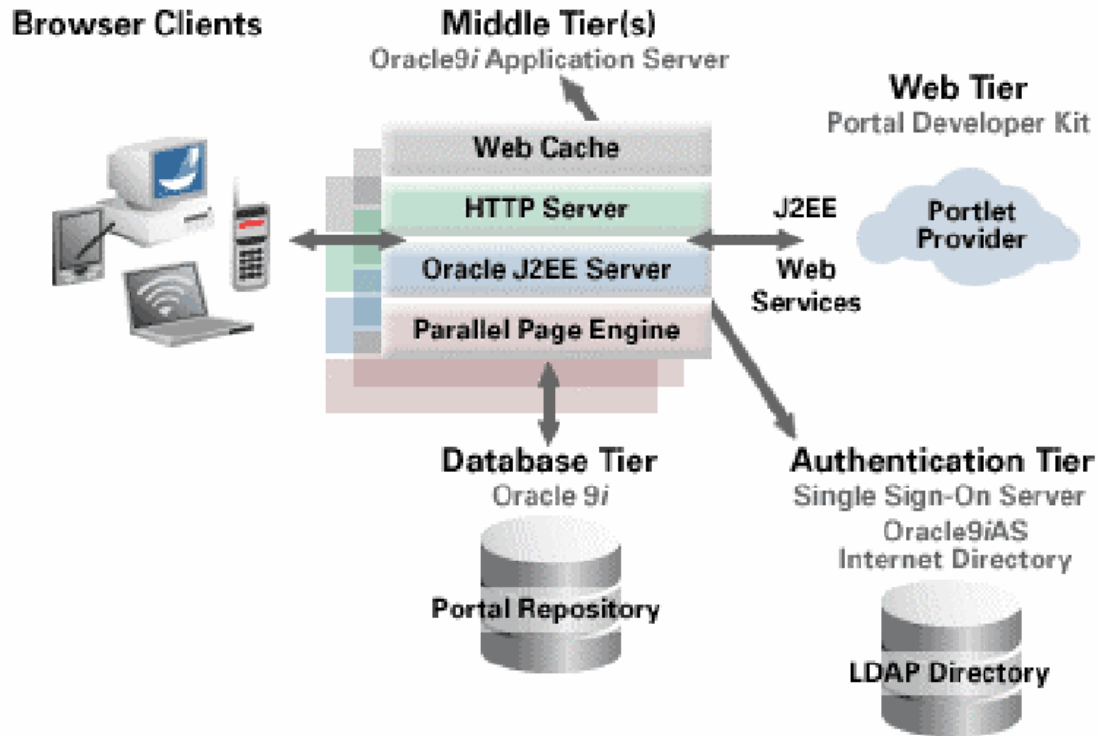


Figure 8. Portal Architecture-*Oracle9iAS* (From Oracle, 2002a)

A third portal product is the Oracle9iAS Portal (Oracle, 2002). It operates from a higher perspective than those previously discussed and has the features of open-architecture interoperability, scalability to meet performance requirements, essential application and content integration, a flexible management model to simplify administration, and productivity tools for portal creation and maintenance. Figure 8 displays the Oracle9iAS Portal architecture.

The Oracle9iAS Portal architecture supports the integration of remotely hosted applications through open Internet standards such as Simple Object Access Protocol (SOAP), Hyper Text Transfer Protocol (HTTP), and eXtensible Markup Language (XML). The Portal uses portlets to incorporate the use of Java Server Pages (JSPs), Java Servlets, and Enterprise JavaBeans (EJBs) into the portal without the requirement of writing additional code. In addition, portlets provide the ability to integrate proprietary

technologies like Lotus Notes, Microsoft Exchange and client/server applications. Portlets are small-embedded portal components that promote, summarize, or grant access to an information resource. All portlets are role based and are integrated to leverage single sign-on for access. Tools such as Oracle Application Interconnect, Workflow and Adapters are used to access, transform and expose higher-level data from third party applications such as SAP, PeopleSoft or Siebel into a portlet-ready format.

For interconnectivity of locally hosted components, the Oracle9iAS Portal uses open Internet Standards such as Java 2 Platform Enterprise Edition (J2EE), Web Services, JavaScript, XML and other languages. It also incorporates standards-based capabilities that support the Web-based Distributed Authoring and Versioning (WebDAV) protocol, HTTP, and wireless protocols.

Applications and information sources, denoted as portlets, use an entity called a provider to communicate with the portal. Each portlet has a one-to-one correspondence with a provider. After a provider becomes registered with a portal instance, its component portlets become available for placement on a portal page. Providers may be grouped into logical groups regardless of physical location to organize and optimize multiple provider registration. In addition, providers may be accessed in different ways within the Oracle9iAS Portal architecture. The Web Provider may be coded for access with the portal in any computing language (Java, C, C++, etc) or it may be accessed through referencing and optionally employing an eXtensible Style Language (XSL) transformation to the applicable URL or Web Services Description Language (WSDL) document. The Portal Engine and provider communicate via SOAP and HTTP. This allows firewall to isolate remote providers from the portal's middle tier. To support a PL/SQL provider, an adapter receives the SOAP call and then subsequently calls the applicable PL/SQL API.

The Oracle9i Application Server is available on UNIX platforms, including Solaris, Linux, HP-UX, AIX, and Compaq Tru64 as well as Windows NT/2000. It uses the Apache Web server as the HTTP server. In addition, it has a built-in database provider that generates MobileXML to respond to mobile requests. It has an Integration component that contains the Oracle Application Interconnect, a product called Workflow,

and adapters to existing applications such as PeopleSoft, SAP, MQ Series, Siebel and others. A level of abstraction from application APIs is provided by the Integration server and its packaged adapters. Oracle Workflow (another service provided by the Oracle9i Application Server) allows portals to model, automate, and continuously improve business practices through defined rules that govern the routing of information within the server's service scope.

The Oracle9iAS Portal Development Kit (PDK) allows developers to build portlets to handle customer-specific applications or content as well as enforce security with a single-login feature. In addition, developers can capitalize on provider and portlet API-level services for development convenience. To support rapid portal development, the Oracle9iAS Portal uses “an easy to use browser-based, wizard-driven, declarative interface” that enables the portal creator to create most of the portal if not all of it. The Oracle9iAS Portal contains page design and development features that “give administrators, page designers, and end users a powerful environment in which to create content rich, secure, portal pages” without any required programming. Page content may have both portlets and content items. Page templates provide a mechanism to enforce predefined page layout, style, and security settings for portal content. Pages are organized within page groups. These groups may be used to “create ad hoc or carefully controlled content taxonomies.”

A basic component of a Portal page is a content item. Oracle9iAS Portal items are rooted within their item type and may only be created by users with appropriate privileges. Item types define content and attribute information. These attributes are custom fields that contain additional item or page data. Policies for content types, type attributes, and page management within a page group are set by Portal administrators. File-type item content may be published via the Web-based Distributed Authoring and Versioning (WebDAV) protocol to the portal repository. Portal page groups may be mapped as Web Folders using a WebDAV client like Windows Explorer. The Oracle9iAS Portal uses a highly tuned, multi-threaded servlet engine to do parallel processing of portlet content retrieval, cache management, and portal-page assembly and delivery. Portal administration, development, and end-user functions are accessed via a portlet. The underlying architecture uses a fully integrated, intelligent cache to enable

high levels of performance and minimize unnecessary page and content regeneration. In addition to this, the architecture supports “load distribution and parallel execution of portal components across multiple servers.”

The Oracle9iAS Portal uses an integrated security infrastructure to “authenticate users, support single sign on (SSO), and manage users/user group information.” All users are authenticated through the use of Oracle9iAS SSO. With SSO support, users need only login to the Oracle9i Application Server once to obtain access to any SSO-enabled application that they are authorized. To support the underlying security infrastructure, Oracle9iAS SSO is completely integrated with the Oracle Internet Directory (OID). The OID is the repository for Oracle9iAS Portal's user and group definitions. User administration is achieved through user management screens in the Oracle9iAS Portal, or through the OID provided APIs and administrative tools. The OID can synchronize with LDAP directories using built-in meta-directory capabilities. For finer granularity of control, user and group privileges on portal objects (pages, styles, items, portlets, etc.) are managed through access control lists (ACL's). With ACLs, responsibility can be delegated by administrators to object owners, who then can specify users/groups and their object privileges. This enables administrators to grant global privileges to all objects of a particular type. In addition, the portal can be configured to provide Secure Socket Layer (SSL) connectivity with users and remote portlet providers.

Oracle addresses the issue of shared versus dedicated databases through its virtual private database. The Virtual Private Database (VPD) includes application context and finely-granular access control. With VPD, access policies for individuals or groups can be defined within the context of the organization to which they belong. Access can be accomplished by providing authentication information either directly or indirectly through a branded Universal Resource Locator (URL). Branded URLs switch the access context to the appropriate subscriber by pointing to a specifically correlated URL linked to the host portal “so that users need only provide their username and password to login.” Branded URLs also give subscribers the ability to include content and public pages on a particular portal.

D. PLUMTREE

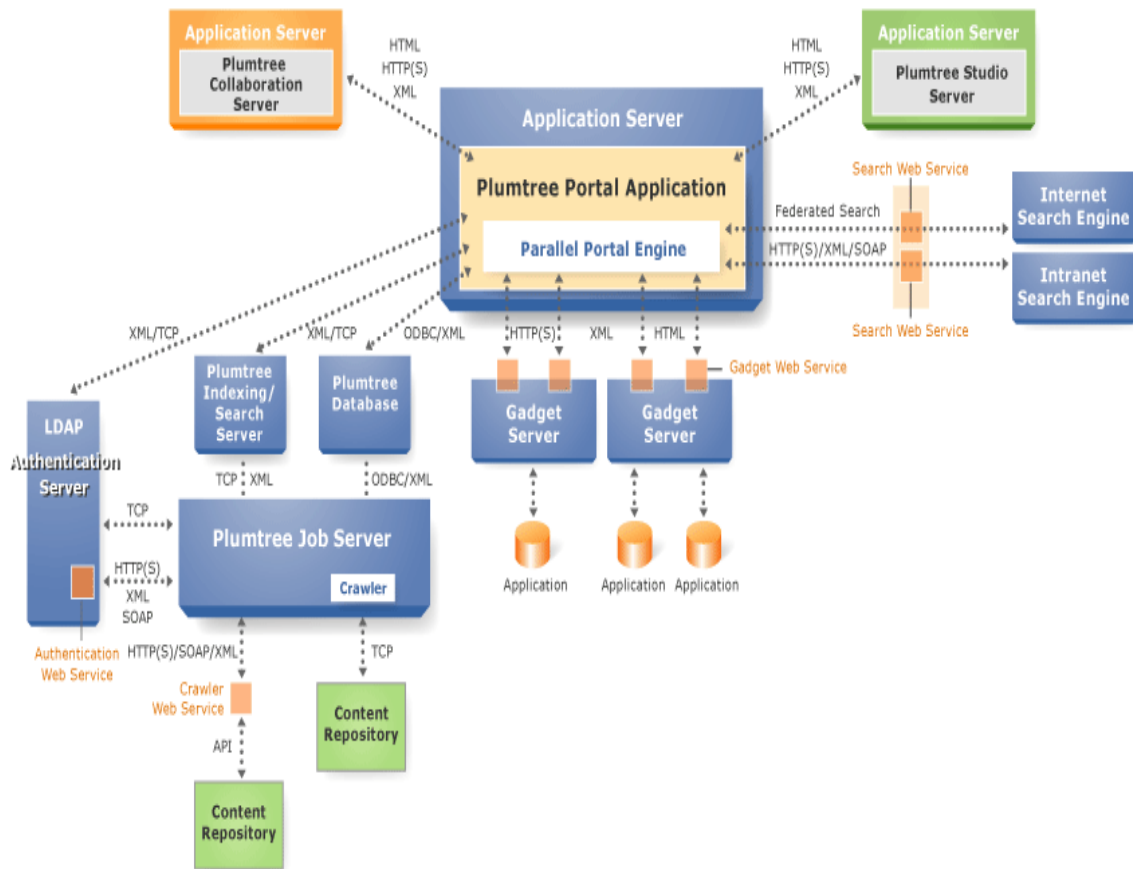


Figure 9. Plumtree Portal Architecture (From Plumtree, 2002)

The Plumtree Corporate Portal (Plumtree, 2002) incorporates heterogeneous applications, security, and content search as Web services by communications between software components and a parallel engine. This infrastructure relies on the Simple Object Access Protocol (SOAP) Web services standard which sends embedded programming commands in an XML text message through HTTP to Web services. SOAP was designed for programmatic interaction between Web service components via a programming interface. This enables the assimilation of different resources as well as scalability for increased user populations. Figure 9 displays the Plumtree Corporate Portal architecture.

Plumtree's Web Services Architecture for integrating heterogeneous technologies in a portal is flexible enough to enable Web services to be "installed anywhere on a wide area network, on any platform, using any programming language to communicate with

other systems.” Through the appropriate SOAP implementation and any programming language, developers can create Web services or applications that use them. This means that different environments and networks may be used to create and operate Web services and clients while still supporting interoperability and common communications over HTTP and SOAP. Service flexibility is essential to almost any size portal deployment. Because Web services are modular, the integration and interoperability of portal components provided by different vendors is possible. The problems with integrating portal components provided by different vendors have not been with the connectivity between components but with the content of the messages.

The Plumtree Corporate Portal depends on Web services to perform all functions that communicate with portal-incorporated systems. This allows the extension of core portal server software without redevelopment. It enables an organization to add new applications, index new content, authenticate new users, and search new repositories by developing and registering these new modules as Web services. Authentication, searching and indexing Web services receive commands as SOAP messages from the portal server. Subsequently, they provide the appropriate response to the portal server. Gadget Web Services, on the other hand, handle user interactions through a user interface. The portal server only sends the user’s initial request to the Gadget Web Service which resides on the gadget server. This request includes user or group identity and preferred configuration information to personalize the Gadget Web Service for that user or group. The Gadget Web Service functions respond to user requests provided through the gadget’s user interface; no further commands from the portal server are provided. It is important to note that Gadget Web Services are more like “modular Web applications that the portal service can combine in a portal experience.”

Each Gadget Web Service can have separate preferences tied to a separate portal-registered interface for configuring them. Because the HTTP header delivers user preferences, preferences are more quickly processed than SOAP message commands. This is an advantage because gadgets dynamically incorporate several applications in a single user session. For example, a simple Gadget Web Service providing a calendar display may call components for authentication, calendar display, and new appointment creation, among others. Providing components as Web services enables the Gadget Web

Service to use them and simplify gadget development. However, significant effort is required to build them.

The Plumtree Corporate Portal solution uses a parallel portal engine to improve performance and integrate resources using accepted Internet standards such as HTTP, SOAP, and XML. The parallel portal engine operates within an application server and uses the Plumtree HTTP library to issue multiple simultaneous HTTP requests. These requests are handled through one computing thread in a single “virtual request” to many Web services. This reduces the number of network sockets opened and closed and allows load balancing against multiple instances of a Web service by redirecting requests when the service is overtaxed or fails. To further increase processing speed, the addresses of computers hosting portal web services are cached in memory by the parallel engine. In addition, the parallel engine supports a secure socket layer (SSL) implementation that is “optimized for secure connections to thousands of Web services.”

The Plumtree Corporate Portal provides flexible authentication, granular access control, and adaptability to host systems with diverse security schemes. Authentication in the Plumtree solution is achieved through its own master user directory (password-based authentication) or through the use of external services that enable a single sign-on to multiple network systems. The Plumtree Corporate Portal creates a master directory of users that includes any group affiliations from external services. Within the context of the master directory are user-associated profiles with “portal-specific privileges and personalization preferences.” This master directory of users and groups acts as a master access control system for all Plumtree Portal links. Consequently, the master user directory can negotiate between the various incompatible user directories of enterprise applications.

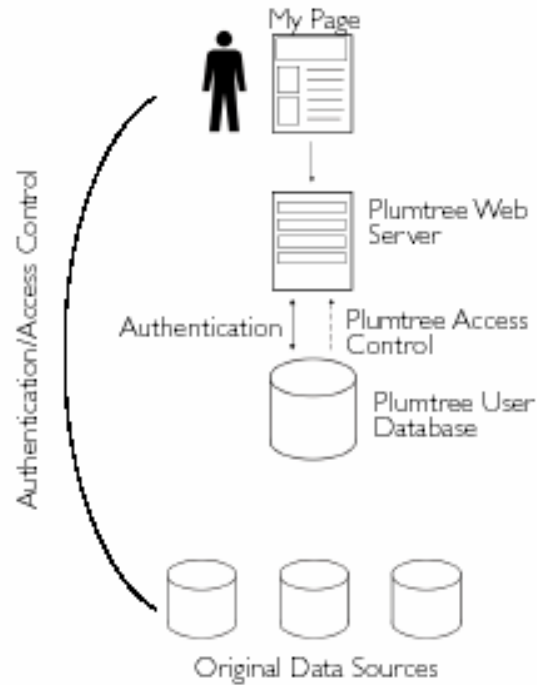


Figure 10. Plumtree Password-Based Authentication (From Plumtree, 2000)

Figure 10 displays the typical password-based authentication for the Plumtree Corporate Portal. All Plumtree users have an account whose information is encrypted in Plumtree's back-end database. Users access a Plumtree Web page and type in their user name and password. The login is authenticated against the master user directory database. When properly authenticated, users enter the portal possessing. "Security permissions as defined entirely within Plumtree's user database."

Figure 11 illustrates how the portal authenticates users with an external directory service to process the authentication. To ensure that access control updates are received quickly, the Plumtree master user directory database is periodically synchronized with appropriate external directory services. When databases differ, user and group information from the external directory service is imported into the master user directory database.

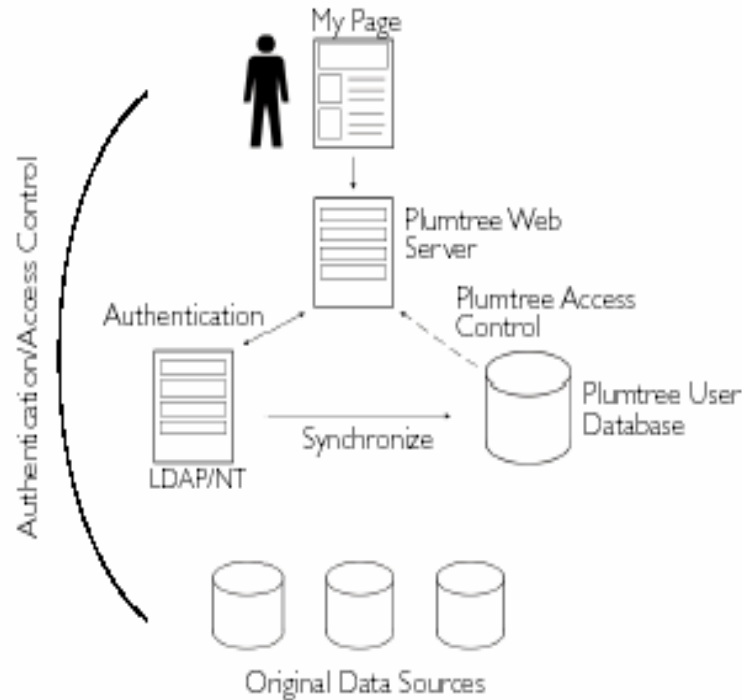


Figure 11. Authentication by External Directory Service (Plumtree, 2000)

Plumtree supports secure client authentication as well as single sign-on to multiple applications through network authentication standards such as Kerberos and X.509 v3 digital certificates. Kerberos authentication uses secret-key cryptography and is based on the idea of tickets, “encrypted data packets verifying a user’s identity that are issued by a trusted authority called a Key Distribution Center (KDC).” During a typical Kerberos experience, a user opens a Plumtree web page and performs the standard log-in procedures. The user is authenticated by the Kerberos KDC. Once properly authenticated, the KDC provides an initial Ticket Granting Ticket (TGT). When the user navigates through the portal to a network resource that requires user authentication for access, the session presents the TGT to the Key Distribution Center and requests the issuance of a Service Ticket (ST). The user’s session presents the ST to the network resource and is granted the appropriate user access.

X.509 is a standard for defining digital certificates which enable Public Key Infrastructure (PKI) based authentication. PKI uses public and private cryptographic keys to authenticate a user. These keys are “mathematically related, yet impossible to

deduce from one another.” Under PKI, a trusted service called a Certificate Authority (CA) is used to verify user identity and manage digital certificates (user credentials). During a typical PKI transaction, a user opens a Plumtree web page and performs the standard log-in procedure. The Plumtree server generates an encoded certificate request to the CA using the user’s private key. When the CA receives the user’s certificate request, it extracts the user’s name and authenticates the user’s request by decrypting the transmitted certificate request using the user’s public key. When the certificate request is authenticated successfully, the CA will issue a certificate associated with that user to the Plumtree server. The user’s certificate is proof of user authentication for that particular session. When the user navigates through the portal to a network resource that requires user authentication for access, the certificate is presented and the appropriate user access is granted.

The Plumtree Corporate Portal uses a role-based security model and user groups to provide access control. All users are members of at least one group, and each group is related to a role. Each system object has an access-control list (ACL) that determines user privileges for the object. The Portal supports three user roles and up to three privileges for each system object. A combination of privileges, roles, and groups determines user access and rights.

Table 2 summarizes user roles and the specific privileges they may exercise according to a particular system object. Administrators or Content Managers can change group privileges; however, by default certain roles assume specific privileges. Groups can have different privileges in different parts of the portal. This is the key to Plumtree’s granularity of access control. For example, a supply expert may be a Content Manager in a specific subsection of the portal but they may only have read access in another area of the portal.

| | Content Managers | Content Maintainers | Browsing Users |
|-----------------------|--|---|---|
| Links | Read Only: Can see the link Read/Write: Can see the link and modify any of its properties | Read Only: Can see the link Read/Write: Can modify the link's name and description | Read Only: Can see the link Read/Write: Can see the link |
| Folders | Read Only: Can browse the folder Read/Write: Can submit, move and remove links, move, delete, or modify the folder Read/Write/Approve: Can add and remove links, and approve links submitted by users or imported by crawlers | Read Only: Can browse the folder Read/Write: Can submit, move and remove links, move or delete folder Read/Write/Approve: Can add and remove links, and approve links submitted by users or imported by crawlers | Read: Can browse the folder Read/Write: Can submit links to the folder, pending approval Read/Write/Approve: Can submit pre-approved links into the folder |
| Publications | Read Only: Can see the publication Read/Write: Can modify or delete the publication and approve its issues | Read Only: Can subscribe to the publication over the Web Read/Write: Can approve issues of the publication | Read Only: Can subscribe to the publication over the Web Read/Write: Can subscribe to the publication over the Web |
| Gadgets | Read Only: Can use the Gadget and personalize it where applicable Read/Write: Can modify or delete the Gadget | Read Only: Can use the Gadget, and personalize it where applicable Read/Write: Can use the Gadget, and personalize it where applicable | Read Only: Can use the Gadget, and personalize it where applicable Read/Write: Can use the Gadget, and personalize it where applicable |
| Data Sources | Read Only: Can see and crawl the data source, and create links to information in the data source Read/Write: Can modify or delete the data source | Read Only: Can submit links to information in the data source | Read Only: Can submit links to information in the data source |
| Properties | Read Only: Can see the property's values Read/Write: Can modify the property's values or delete it | Read Only: Can see the property's values | Read Only: Can see the property's values |
| User Groups | Read Only: Can see the user group Read/Write: Can add and remove members | N/A | N/A |
| Crawlers | Read Only: Can see the crawler Read/Write: Can modify and delete the crawler | N/A | N/A |
| Document Types | Read Only: Can see the document type Read/Write: Can modify or delete the document type | N/A | N/A |
| Jobs | Read Only: Can see the job Read/Write: Can add or remove operations, modify the job's schedule, or delete the job. | N/A | N/A |

Table 2 Default User Roles and Privileges to Objects (From Plumtree, 2000)

The Plumtree Corporate Portal uses portal-object ACLs for widespread extended enterprise deployment of the portal to users with varied security profiles. The portal recognizes four different objects: Links, Folders, Gadgets, and Publications. Plumtree defines a link as an indexed content directory element that points to external information. Folders are defined as content directory elements that contain links. Gadgets are portal-embedded applications or services. Publications are personalized page-embedded content

directory queries or queries received via email. All of these objects are secured with an ACL. This means that if a user does not appear within an object's ACL or is not a member of one of the groups listed within the ACL, the user cannot see the object nor search for the object. Finally, if the user does not have the appropriate document access in its native environment, then it will also not be accessible through the portal. Figure 12 summarizes the four access control checks for portal objects.

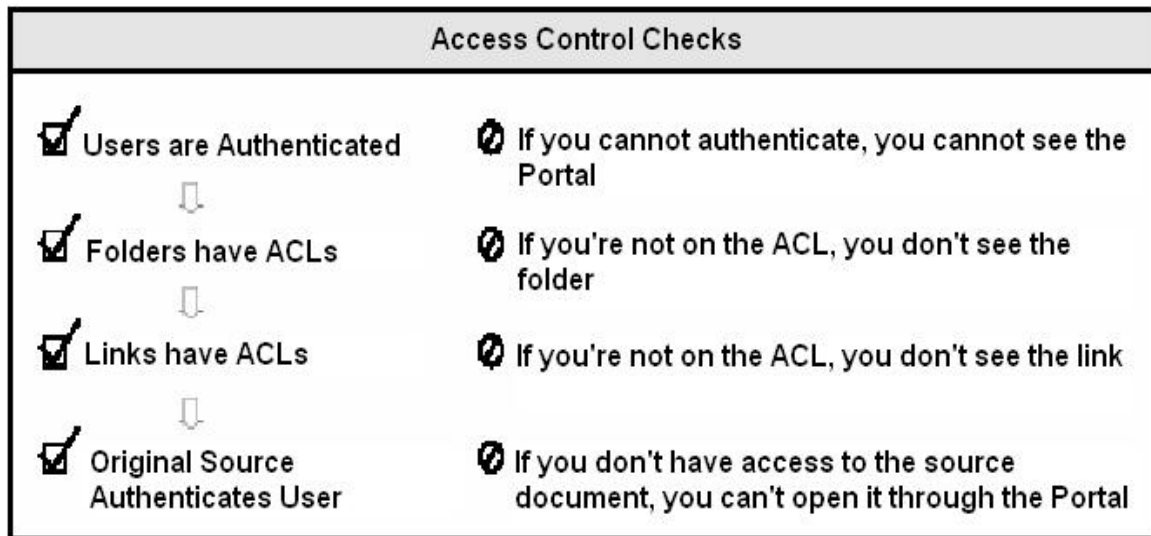


Figure 12. Four Access Control Check Points (From Plumtree, 2000).

Plumtree's Corporate Portal can work with the security schemes of numerous hosts. This allows the portal to act as a conduit or gateway for information and applications from disparate systems. To support this effort Plumtree crawlers "catalog information from host systems" and Plumtree gadgets "embed Web application modules that require authentication with a host." This results in the portal managing multiple authentication settings to ensure that information is only received by its authorized intended recipients.

Plumtree uses a content directory that is created from information from several host systems, namely databases, file systems and web sites. To create this content directory, the portal must access and link disparate original data sources. Often these data sources use differing methods to manage objects and users. This creates a challenge for integration. To address this Plumtree employs a three-layered security scheme to

“empower Content Managers to access the resources necessary to create a comprehensive portal.” Content Managers can:

- Provide each crawler with a particular security profile. This limits the crawler to only the authorized information on the host system.
- Instruct crawlers to stamp any created links with Plumtree ACLs corresponding to the profile used to access the information.
- Prevent other Content Managers from using or editing their crawlers.

In addition, Administrators and Content Managers can configure the Portal Server to operate as “different users when polling different data sources for information.” This allows the portal to manage information links that reference original data while it remains in its “native platform, safeguarded by its native security system.” Since the format for ACLs may vary between systems, the portal does not import host system ACLs. Instead, Administrators and Content Managers are able to configure individual crawlers to stamp links with a Plumtree security profile that matches the “security profile used to access the host system.” This provides the foundation for the accessibility of disparate objects that are embedded as portal content.

Merging big applications into a portal page at user discretion challenges enterprise security infrastructure in two ways: Who can access which modules, and how do disparate modules authenticate in the context of their host systems? Through the use of common ACLs associated with each application module, Plumtree Administrators and Content Managers can control which gadgets may be embedded into personalized pages and by whom. Different functionality provided by a gadget can be managed with different ACLs. For example, a budget gadget from a database management service may be restricted to personnel who deal with budgets, while an equipment-inventory gadget can be made available to everyone in the supply department.





| |  ← Domain Users |  ← Managers  ← Executives |  ← Executives |
|---|--|--|--|
| Crawler 1 Runs as Domain User. Stamps Employees ACL. | Imports. Stamps Employees ACL. | ⊘ Doesn't read. | ⊘ Doesn't read. |
| Crawler 2 Runs as Manager. Stamps Sales Managers ACL. | Imports. | Imports. Stamps Sales Managers ACL, Sales Executives ACL. | ⊘ Doesn't read. |
| Crawler 3 Runs as Executive. Stamps Sales Executives ACL. | Imports. | Imports. | Imports. Stamps Sales Executives ACL. |
| Final State | Link has Employees ACL. | Link has Sales Managers, Sales Executives ACLs. Invisible to others. | Link has Sales Executives ACL. Invisible to others. |

Table 3 Crawlers Used To Import Increasingly Sensitive Information From A Host System. (From Plumtree, 2000)

It is important to note that gadget ACLs do not provide a mechanism to conduct authentication between the host system and the gadget. Plumtree handles this by capturing “users’ authentication information for each embedded application or service” and maintaining this information as encrypted entries in its user database. This provides users access to important parts of embedded applications without the need for separate authentication.

Some Web-based application use persistent “cookies” to provide users with personalized information every time they visit a site. This creates an impediment for systems that rely on servers to extract information from that Web site as a persistent cookie is expected to be with the client. Plumtree manages cookies on the server. This allows the server to pass the appropriate cookie to the external Web site, obtain the appropriate information, and assemble it on the user’s tailored portal web page.

E. RELATED RESEARCH

Designing security into information technology from the onset of the project is not a novel concept as evidenced from the above commercial portal examples. Most information assurance experts will agree that designing and implementing security mechanisms and processes at the onset of an information technology project is the best way to proceed. In the case of vortals, many design teams assume that they can design security into their vortal solution through a thorough threat analysis. Subsequently they use this information to develop specific system requirements.

Although this may provide a great deal of information concerning the existing threats and vulnerabilities, what happens when the vortal persists over a period of time and faces new threats? Will a threat analysis be conducted for every small system change? Who will make the decision to conduct these threat analyses and by what criteria will these decisions be made? The design approach and implementation can frequently provide a clear engineering direction. Some related work in the field of research has produced unique approaches that can be considered in the design methodology.

Grupa gives guidelines for forming information-security policies and survival strategies in a dynamic and hostile business environment (Grupa, 2001). He examined an artificial economy within a synthetic environment and ran simulations to test and evaluate strategies to counter threats. This research was for online financial services provided by organizations such as banks. It tests security policy implementations by having human actors select security policy configurations and try simulations to test these configurations.

McDermott and Fox describe a method which is a variation of Universal Modeling Language (UML) Use Cases, for the purpose of developing security requirements of an information-technology system in a simple way (McDermott and Fox, 1999). Through the use of an “Abuse” case, in which a use case is adapted to capture and analyze security requirements, McDermott and Fox can provide a more detailed description (resources, skills, and objectives) of the actors within a specific use case

scenario while avoiding mathematical security models. Thus they make security requirements easier to understand.

Brewer describes lessons from large-scale service providers in the form of a framework for the development of high availability, evolution, and growth within web portals and Internet service providers such as AOL, Microsoft Network, and Yahoo (Brewer, 2001). The conclusions presented take the form of principles and recommended approaches rather than a quantitative analysis of methodologies for the development of high availability, evolution, and growth.

Kargl describes a means of preventing distributed denial-of-service attacks to Web sites based on a class based routing mechanism in the Linux kernel rather than on tight security policy and third-party software and hardware mechanisms (Kargl 2001). He provides a scenario, defines denial-of-service and distributed denial-of-service attacks, classifies the defined attacks, and discusses a recommended Linux kernel solution to prevent these types of attacks.

III. ANATOMY OF A VORTAL

A vortal is similar to a personal computer operating system with a graphical user interface (“GUI”). Many popular operating systems such as Redhat Linux®, Sun Solaris®, and Microsoft Windows® use a graphical user interface (Machiraju, 1996). Operating systems are software packages that provide an abstract mechanism to manage and present services and functions necessary to manipulate data, software applications, and services provided by an information system. They are software applications that manage the overall computer resources and services by acting as an intermediary between the user and the computer hardware (Silberschatz and Galvin, 2001). Similarly, vortals act as intermediaries between the user and required resources and services. In fact, vortals act as management packages configured to provide, in a graphical manner, the required functionality and capabilities of various resources and services in a network of computing resources.

A. THE INFORMATION ENVIRONMENT

Typically, the information environment with which a vortal user must navigate is hybrid in nature. Hybrid means that it presents a range of heterogeneous information services in an integrated and consistent manner. Service provision variations allow disparate service and business models to exist within such environments. Additionally, they support “some or all of the following functions: discovery, location, request, delivery and use, regardless of the domain in which objects are held” (Russell, 1999).

Good hybrid information environments feature transparency to the user as well as flexibility and scalability. Transparency requires the environment to depend on underlying protocols, software, and systems. It should permit the inclusion of new components as new requirements emerge and as new standards develop. Open-standards “pluggable” components that can seamlessly interface with others facilitate “services for discovery, location, request, delivery, and use.” In addition, good hybrid environments should ensure services to be provided in a consistent manner across multiple domains irrespective of medium, and should be conducive to growth in “both service provision and in volume of traffic.”

Consistent management of the hybrid information environment requires an organizational and technical framework for which systems and data providers as well as information and statistics as well as obtaining components and setting standards for interoperability. Environment management should not require significant expertise and resources. Additionally, the environment should handle both trusted and untrusted users and resources, and should support authentication and configurable access for everything in the environment.

B. ABSTRACT MODEL

Typically, an abstract model describes architectural objectives. We use the MODEL Information Architecture (MIA) (Gardner, 1999).

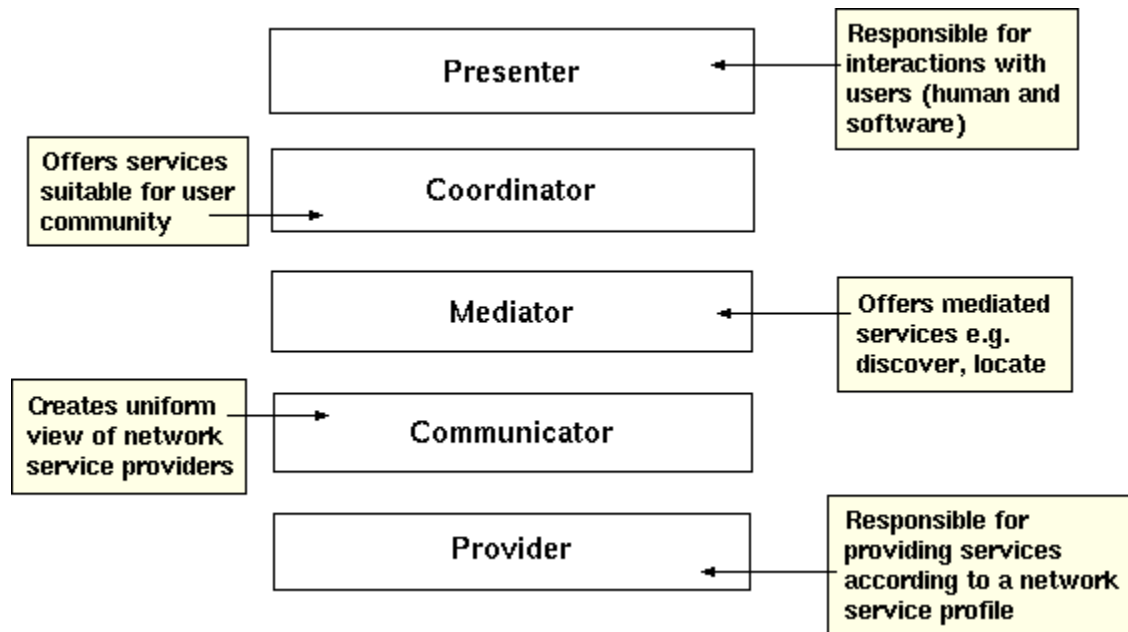


Figure 13. Five Layer MIA (From Gardner, 1999)

Figure 13 illustrates the five-layer approach of the MIA. This achieves simplicity while supporting complex functionality: Each layer can have its own abstractions and purposes. The central Mediator layer registers the various services offered but not their implementation differences. The Communicator layer provides the Mediator with standardized access to the offered services. The Coordinator layer “tailors the information landscape for a particular user community.” The Presenter layer communicates with users (people or software agents) to ensure that multiple interfaces

can be supplied to the same Coordinator. Finally, the Provider layer contains the external services provided to the environment.

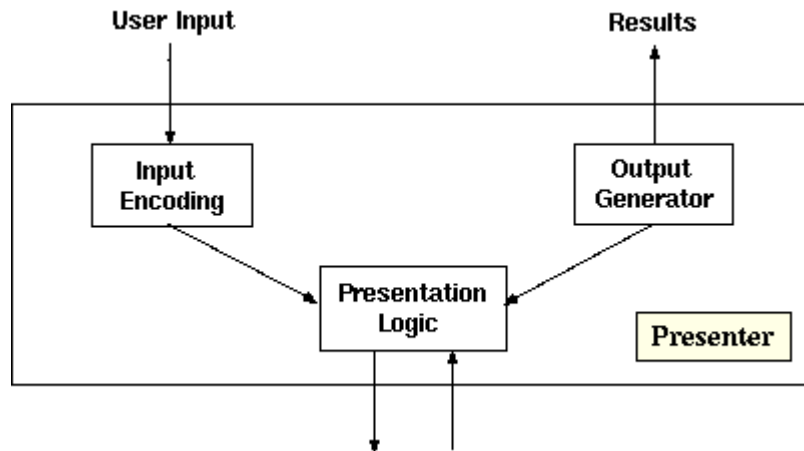


Figure 14. Presenter Layer (From Gardner, 1999)

Figure 14 displays the Presenter layer that manages user interaction. Typically, user interaction involves a GUI or a network protocol. For example, user input submitted via a web form is encoded and passed to the presentation logic. If the input can be handled within the presenter layer (e.g. change of display options) then it is passed to the output generator which updates the display. If the input cannot be handled locally (e.g. a search request), the input is passed to the Coordinator layer and then to the Presenter for display update. This enables the Coordinator to focus on application issues independent of the manner in which a system is accessed.

The Coordinator layer provides an abstract layer for handling applications. It is responsible for managing the “user landscape” and for protecting the Mediator layer from “user-specific and context-specific issues.” Figure 15 displays the Coordinator layer. The User Profile and Session Control contextualize the request. If the request can be handled within the Coordinator layer it is resolved (e.g. current query’s next series of results). More often, these requests will be passed to the Mediator layer whose result will also be contextualized (e.g. select the current file or database record).

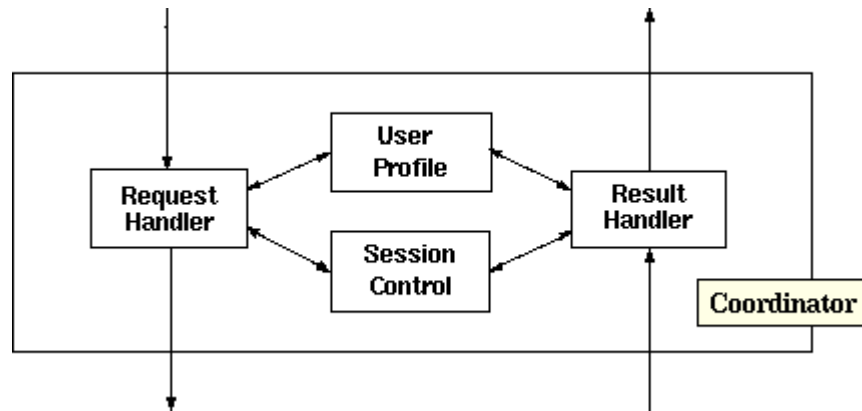


Figure 15. Coordinator Layer (From Gardner, 1999)

The Mediator layer provides an abstraction layer between the coordinator and any “brokered services.” The Mediator provides united services “based on multiple individual services.” Figure 16 displays it. The Request Server receives a Coordinator request and determines the request response. If it is a complex request then it may be subdivided into multiple sub-requests. Each sub-request is correlated to a particular service as determined by the Request Server. Forward Knowledge assesses whether the services will respond positively and provides that information to the Request Server. Subsequently, the Request Server makes multiple sub-requests to services over the Communication layer. Results are combined and provided to the Coordinator.

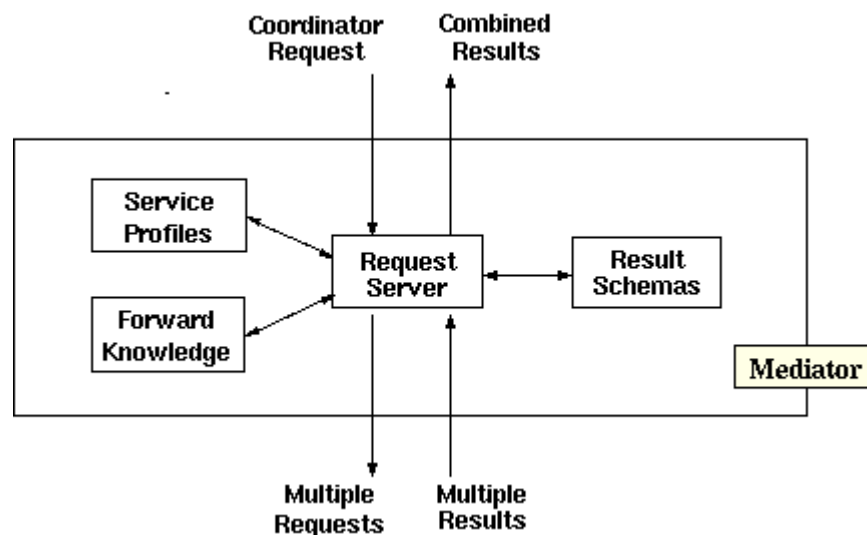


Figure 16. Mediator Layer (From Gardner, 1999)

Figure 17 displays the Communicator layer that manages communication with external services. The Communicator receives multiple requests from the Mediator. These requests are executed in parallel and the results are returned to the Mediator when available. Requests are translated into the proper format based on the Network Service Profile of the target service. The request, including service location, is provided to the Protocol Gateway module for further transfer to the appropriate service. In the reverse direction, when results are received they are translated into a “standard format and vocabulary” for further processing by Mediator.

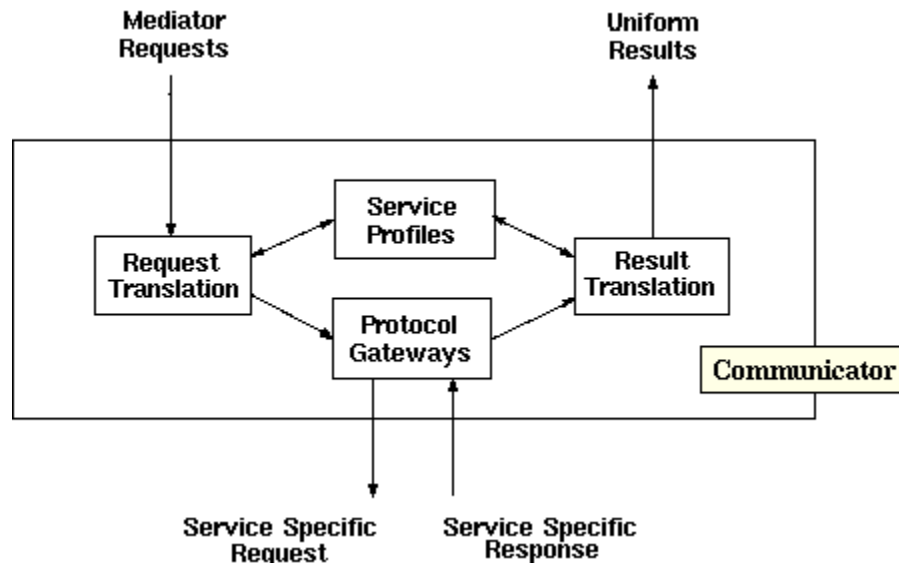


Figure 17. Communicator Layer (From Gardner, 1999)

The Provider layer contains services accessed by the system and brokered across the hybrid Information Environment. Each Provider is described by a Network Service Profile.

C. MODEL APPLICATION

The MIA can be applied to the architecture of a vortal. Figure 18 displays a simple vortal architecture that is broken out using MIA layers. But a real vortal would have more end users and more Provider layer components.

In general, end users (human or intelligent agents) interact with Presenter components. Each Presenter component supplies the server side of an application protocol (e.g. XML, HTTP, HTTP(S) and SOAP) and handles end-user input and output.

Core application logic components within the Coordinator layer interact with the Mediator layer to obtain end-user authentication, personal profiles, tailored display settings, and build the “user landscape” (Powell, 2000).

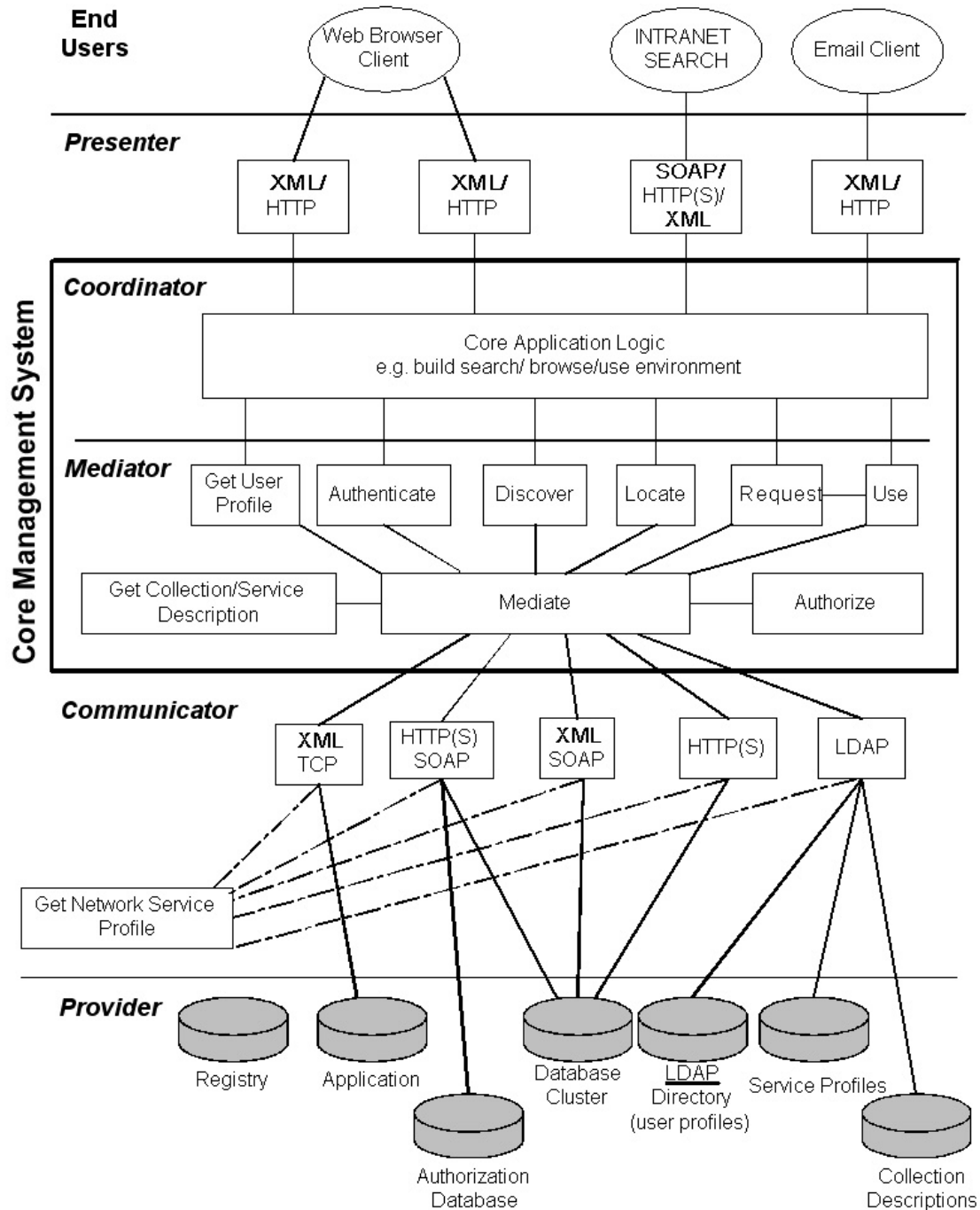


Figure 18. Simple Vortal Architecture (Adapted From Powell, 2000)

For example, to authenticate an end user, components within the Coordinator layer request authentication from the Mediator layer. The generic Authenticate component within the Mediator layer passes the request to the Mediate component which calls on the Get Collection/Service Description component to query the availability of authentication services. The results of the query determine the protocol-specific components within the Communicator layer to be used. The protocol-specific components communicate with the Get Network Service Profile to determine the specific protocol details (port, attributes, etc.) for the desired Provider layer component.

Provisions in the architecture allow for end users to communicate directly with end collections and services as necessary. In addition, vortals can be chained so one vortal can act as the intelligent end user for another vortal.

D. FUNCTIONAL COMPONENTS

If the software that drives a vortal cannot be trusted to adequately provide required capabilities in a reasonably secure manner, then the vortal does not add true value to the organization. In effect, it may even become a liability to the organization by creating risks and exploitable vulnerabilities that greatly exceed the benefits (McGraw, 2002). So its security features are important. To do this, an effective vortal should try to provide capabilities through simple modular components and open-standard communication protocols. These components should be created from what is necessary and sufficient to realize the required capabilities of the vortal (McGraw, 1998). In addition to proper design, configuration control and use based on clearly defined security policy are needed for good security practice. These significantly reduce the likelihood of unintended functionality as well as increase reliability.

So development of a secure vortal requires good software engineering practice. The design of responsibly secure software should follow ten principles (McGraw and Viega, 2002):

- Secure the weakest link
- Practice defense in depth
- Fail securely
- Follow the principle of least privilege

- Compartmentalize
- Keep it simple
- Promote privacy
- Remember that hiding secrets is hard
- Be reluctant to trust
- Use your community resources

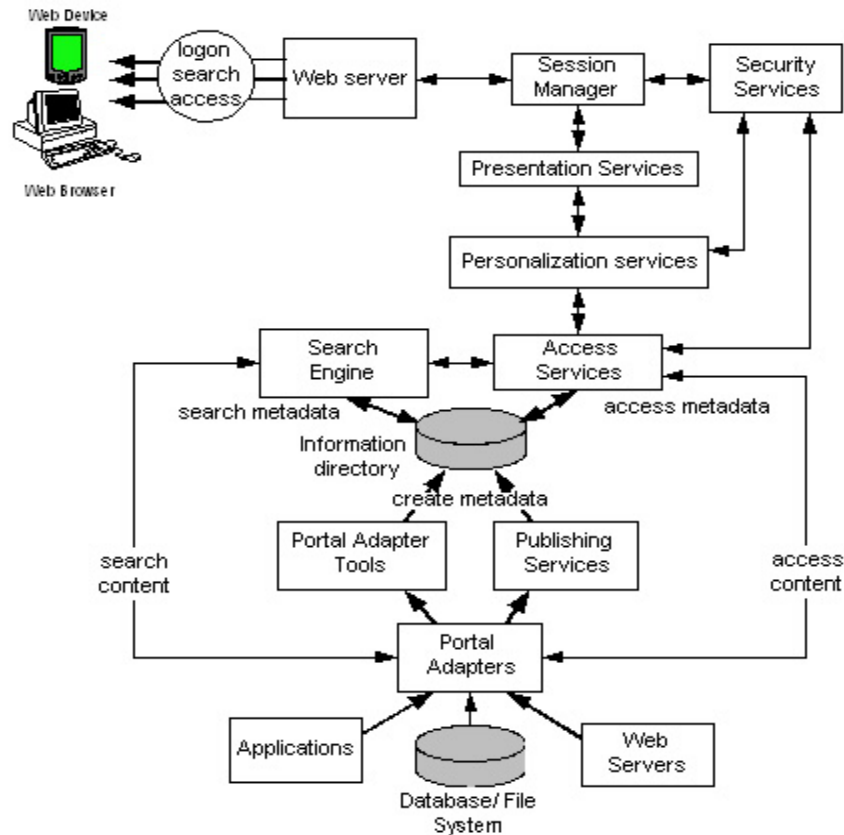


Figure 19. Basic Portal Architecture (From White, 2001)

Figure 19 shows a classic portal design which can be used for vortals. There are six functional components of a vortal: presentation services, core management system, transmission system, data and content, hardware, and additional supporting services. Each of these components could be assigned to a design team.

1. Presentation Services [Presenter Layer]

The vortal interface refers to the elements on the computer display screen, which allow the user to interact with the system. Because a vortal can be presented to a user in

many different ways, measures of effectiveness should be defined for each vortal interface (Hewett, 1992). The target audience for the vortal will prefer mechanisms which more successfully satisfy their needs. Therefore, development should follow good engineering practice with respect to human-computer interaction (HCI). Figures 20 and 21 summarize HCI development.

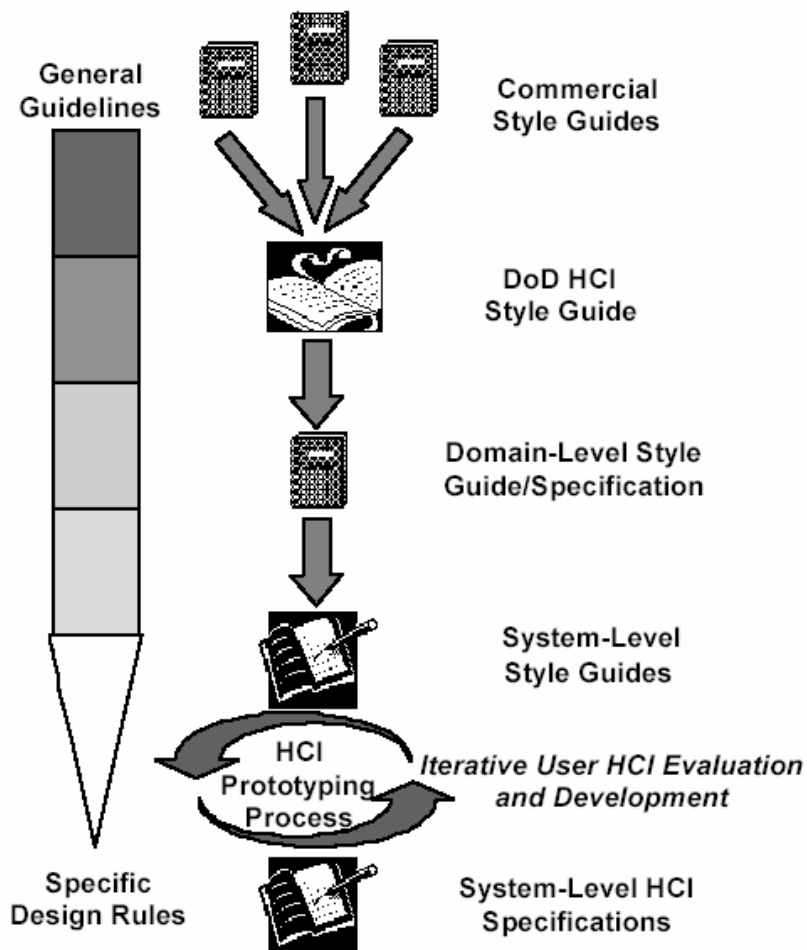


Figure 20. HCI Development Guidance (From DoD, 2001)

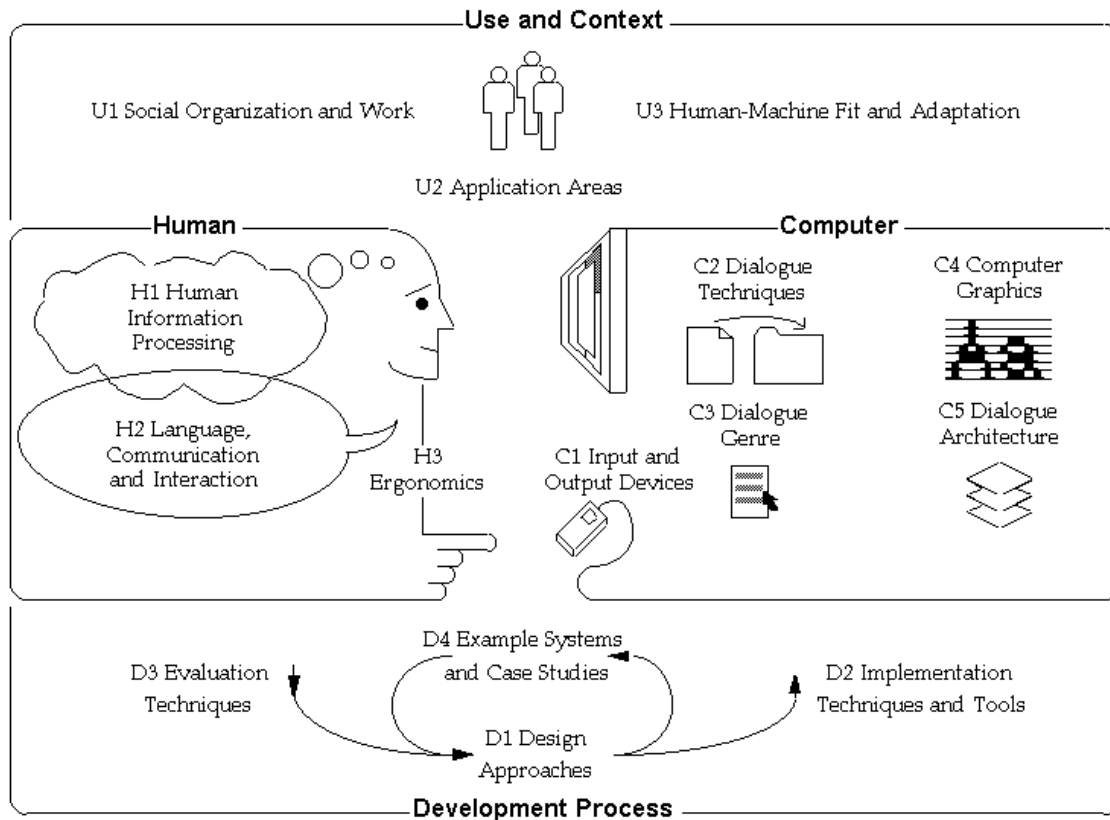


Figure 21. Human-Computer Interaction Process (From Hewett, 1992)

2. Core Management System [Coordinator and Mediator Layers]

The vortal core management system (CMS) is another important component. It includes both management components such as the Core Application Logic and software components. It is possible to use hardware with embedded firmware for components in the Mediator layer. The CMS manages the vortal HCI (“user landscape”) and mediates requests for vortal resources (Russell, 1999).

Decisions to utilize a prepackaged software product or to develop the CMS should depend on both a thorough cost-benefit analysis and the requirements of the vortal. If the value of the content of the vortal is low and the requirements of the vortal are not great, then it may be more cost-effective to use a configurable prepackaged CMS that meets vortal requirements rather than hire a team of programmers to create a tailored CMS. Keep in mind that there are hidden costs beyond the cost of the vortal content. For example, consideration must be given to the costs associated with CMS compromise. Will CMS compromise result in access to a network containing organizationally sensitive

resources? What are the costs associated with the compromise of any network resources that may be accessed through a compromised CMS?

If the most economical approach would be to hire a team of programmers, then it is important to conduct a risk assessment, including threat and vulnerability analyses. It may be worthwhile to develop common criteria profiles to assist in the development of the CMS. These analyses can potentially identify critical discrepancies within the design and implementation of the CMS, and possibly other areas of the vortal, which may go unnoticed during design and implementation. The programmers should also have a good understanding of the organization's functions and practices to develop context for the vortal, and should be required to follow good software engineering practices.

3. Low Level Communications System [Communicator Layer]

The Low Level Communications System is the mechanism for parallel management of mediated requests, and it handles protocols and the metadata vocabulary. It provides information sharing and management and a mechanism to link the vortal data and content together. A standardized conduit for communications, based on open standards, is necessary for interoperability in general (Plumtree, 2000).

4. Content [Provider Layer]

The content of a vortal is the databases, software applications, and other structured and unstructured data files associated with it. Vortals use resources made available through the core management system. Well-designed and implemented database applications can improve the productivity and value of a vortal (Castano, 1995). For example, a poorly designed database application may be unable to fully and consistently extract all valid data from the database for a simple query, or just may be too slow. Web applications are another important part of a vortal's content. They increase the content value of a vortal and increase the collaboration, integration, aggregation and consolidation of data, information and services available through the vortal.

5. Hardware

Hardware is another significant part of a vortal. It includes workstations, and machines that function as web servers, application servers, Structured Query Language (SQL) servers, and file and data servers. Hardware also encompasses the routers,

switches, hubs, bridges, gateways, and modems. Hardware should comply with organizational standards and operating policies (e.g. U.S. Navy's IT21 standard).

6. Additional Supporting Services

Besides the five parts of a vortal, it is necessary to consider administrative training requirements for system-maintenance personnel and configuration management of the elements of the vortal. Configuration management controls how new components are added and old components are removed. We must also consider administration and management of the vortal information environment, and the lifecycle management of the overall vortal system.

IV. SECURE VORTAL DEVELOPMENT

This chapter provides an overview of the development process for a vortal. Figure 22 provides a diagram. We assume, consistent with the discussion in the last chapter:

- Security within the design, development, and maintenance of software is a high priority for senior management (McGraw and Viega, 2002).
- Architecture should embody applicable security policies (Department of Defense, 2001).
- The design and maintenance of software must follow good software engineering, information assurance, and network administrative practices (McGraw, 1998).
- The objectives of the software are specified by the target audience, organizational policies, and the mission of the organization (Department of Defense, 2001).
- Vortal content changes as the needs of the target audience, organizational policies, and organizational mission changes (Collins, 2001).

A. CONCEPT DEVELOPMENT

Vortal concept development is the first step toward development of a vortal. Concept development can use information from three topic areas: high-level policy and doctrine, organizational mission and policies, and user input.

1. Higher Policy and Doctrine

Higher policy and doctrine provide a mechanism from which overarching organizational requirements and high level concept boundaries are developed. And during their review, it may be possible to identify inadequacies within the policy and doctrine that justify the update of these policies and doctrine.

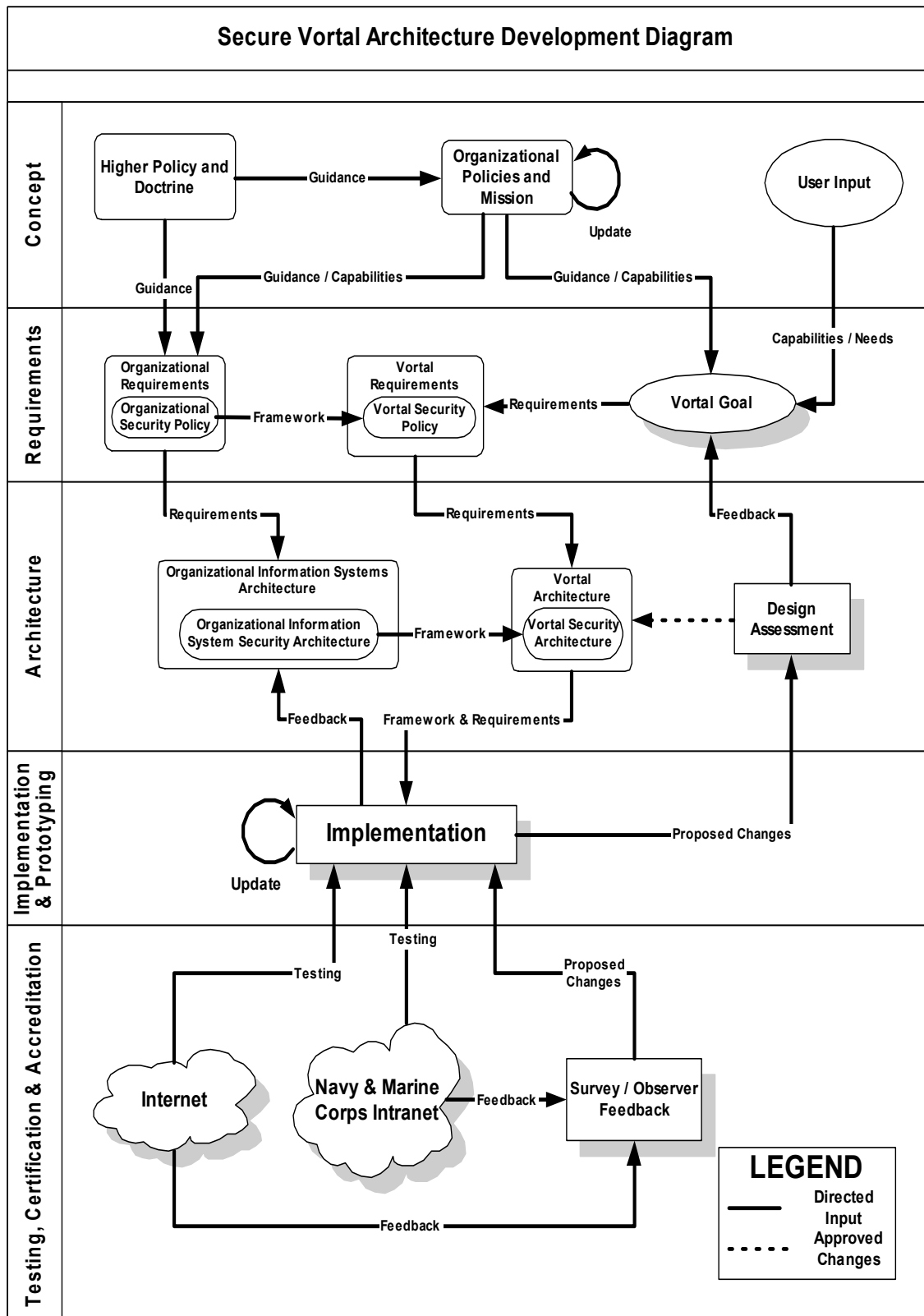


Figure 22. Secure Vortal Architecture Development Process Diagram

2. Organizational Policies and Mission

The organizational mission and organizational policies are derived from higher policy, and doctrine. They provide specific organizational level guidance for the vortal concept development. In addition, organizational policies can be used to refine the vortal goal. The organizational mission defines centers of concentration for the organization and provides necessary organizational focus. The vortal goal is defined with these boundaries and derived from these specific objectives.

In addition, the organizational mission and organizational policies provide specific input for the development of limiting organizational requirements. These limiting requirements can be further refined to form specific organizational information system (ORIS) requirements that are necessary to perform the organizational mission.

3. User Input

User input provides direct, specific contributions that are necessary for the development of the overall vortal concept and subsequent vortal goal. When user inputs are combined with derivatives from the organizational mission and policies, specific requirements can be formed. These requirements may be used to further refine the limits and purpose of the vortal.

User input should reflect a reasonable target-audience sampling and should be analyzed to determine that which supports the organizational mission and applicable policies. Inputs outside the organizational mission or applicable policies may have worth and should be considered in developing possible changes to the vortal goal. The vortal goal should adequately consider extraordinary cases that often help to define unique limiting objectives. Extraordinary cases may lead to “cutting edge” requirements which can help to keep the vortal goal up-to-date and push the vortal to the positive limits of its ability to support the mission and higher policy. Finally, it is also important to note that user input can have not only an impact on vortal goal requirements but also organizational information system requirements that could impact the overall ORIS architecture.

B. REQUIREMENTS DEVELOPMENT

Requirements are essential in the development of a vortal. They codify limitations and set boundaries for vortal development. In addition, they assist in the development of explicit vortal needs and capabilities. Further, they provide guidance throughout the development process, which helps to maintain focus in the development of a vortal. Figure 23 graphically illustrates the relationships between components in requirements development.

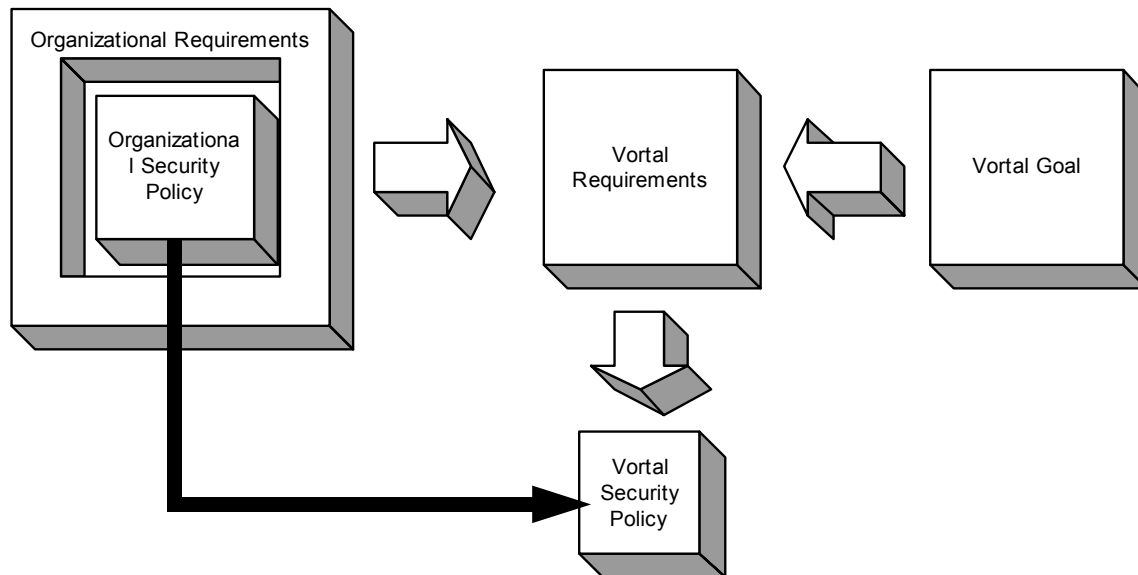


Figure 23. Component Relationships within Vortal Requirements and Security Policy Development

The vortal goal defines what the vortal provides, who it represents, and for whom the vortal provides its content. In addition, the vortal goal helps to determine the vortal's security and other types of requirements, which in turn, directly influence the vortal architecture and content management. Vortal requirements, derived from the vortal goal and the organizational security policy, codify functional and performance needs of the vortal.

The organizational security policy is derived from organizational requirements and shapes the development of vortal security policy. Organizational requirements are derived from higher policy and doctrine as well as organizational policies and the

organization's mission. Functional and performance requirements provide the foundation for organizational security policy and information-system architecture.

1. Organizational Requirements

Organizational requirements provide the framework from which organization-wide information system architecture may be developed. They are derived from the combined inputs of higher policy, doctrine, organizational policies, and the organization's mission. From organizational requirements, the information-processing, transmission, communications and hardware requirements that are necessary to support the organization's mission objectives may be determined. They also provide the foundation for the development of organizational security policy.

2. Organizational Security Policy

Organizational security policy codifies guidance for organizational security requirements. It is developed within the constraints of overall organizational requirements and is based on threats to the mission and the objectives derived from the mission. It provides a general guidance mechanism to ensure that the basic tenets of information assurance are emphasized: "protection from unauthorized or accidental modification, destruction, and disclosure and ensure timely availability and usability of those data (Kabay, 1996)." It provides overarching guidance for the development of specific vortal security policy. The development of organizational security policy follows these steps:

- Preliminary evaluation identifying applicable organizational resources to be protected;
- Management sensitization obtaining approval for an organization-wide audit and policy formulation project;
- A needs or risk analysis analyzing the risks to the resources you strive to protect;
- Construction of policies and procedures that meet the identified needs;
- Implementation (any transformation requires education and may require changes in behavior; it should start at the top with management support); and
- Maintenance by creating and maintaining security awareness through daily awareness in the work environment as well as an annual security agreement). (Kabay, 1996)

3. Vortal Goal Definition

Why do we need a vortal goal definition? This may seem to be a trivial question; however, quite often the development of the answer can be instructive. Every vortal must justify the dedication of resources; it must possess a purpose characterized through the vortal goal.

Figure 24 illustrates the component parts of a vortal goal. The measurement of the vortal's value is predicated on knowing what applications and services the vortal is intended to provide and how effective they were provided. This is accomplished by measuring vortal performance effectiveness with respect to the intended vortal goal.

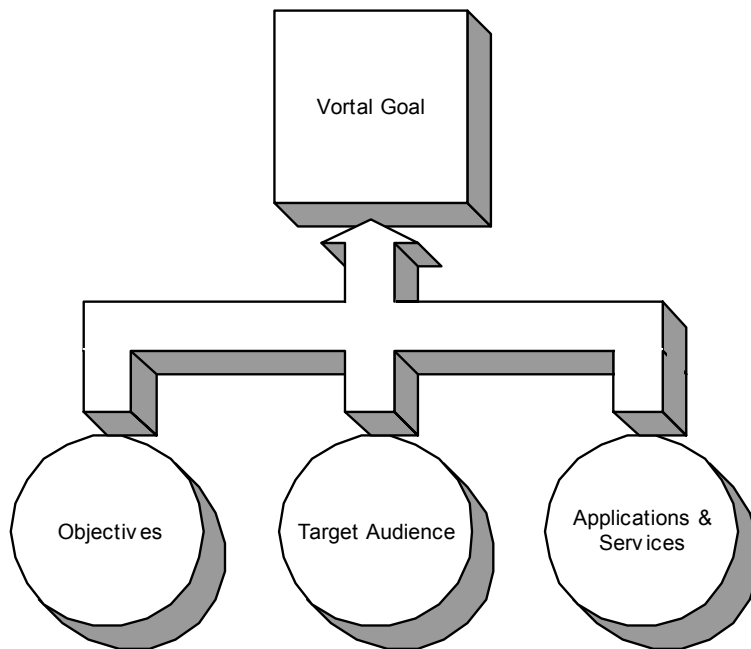


Figure 24. Vortal Goal Components

Within the context of vortal development, the term “goal” has a specific purpose and meaning. It describes what the intended vortal is designed to do; it identifies the beneficiaries; and it provides guidance for the management of data, information, and

services which are available through the vortal. It is primarily used to orient the efforts and actions of the vortal design team.

A vortal design team may have most of the vortal goal definition dictated by higher policy, doctrine, or some other formal document. However, if the dictated portion is not too specific then the design team must obtain further guidance from their higher organization. In some cases, the vortal goal may need to be revisited to re-focus the efforts of the vortal management team and to ensure that the vortal provides that which it was designed to provide. Other vortal goals, which have become outdated, may need to be revisited due to emergent requirements, policy changes, or a reorganization of information resources.

In the absence of guidance from higher organizations, the vortal design team should create a vortal goal definition themselves. This ensures provides forward momentum within the vortal design project, provides clarity for the design solution, and provides a local reference from which to start. Below is a simple example of a vortal goal definition that describes intended vortal purposes, beneficiaries; and guidance for management.

The Information System vortal is representative of the J6 branch of the Commander Third Fleet Staff Organization and provides technical support in the form of computer software updates, computer technical manuals, information technology (IT) feedback processing, and IT trouble call resolution to Third Fleet staff, suppliers, and strategic partners. We provide this support through Internet and intranet accessibility over dedicated fiber optic lines, dialup access, and wireless channels.

A well-written vortal goal definition allows the project management team to align the efforts of every subordinate team within the project, to prioritize the allocation of identified resources, and to focus the efforts of development on the most important data, information, and services provided by the vortal. An effective vortal goal definition answers some fundamental questions such as:

- Who does the vortal represent?
- What specific data, information, and services does the vortal provide?
- Is there any data, information, and services overlap with other vortals?
- For whom does the vortal provide the data, information and services?

- How does the vortal provide this data, information, and services?

Criteria can assess how well written a vortal goal definition is:

- The goal should be clear, concise, and understandable.
- The goal should be brief.
- The goal unmistakably specifies the nature of the vortal, identifying:
 - The vortal's primary target audiences and their needs;
 - Products and services provided to meet the target audience's needs;
 - How primary technologies affect the vortal solution.
- The vortal goal applies to all subcomponents.
- The vortal goal definition identifies characteristics that distinguish the vortal from other vortals which provide the same data, information, and services.

Writing an effective vortal goal definition requires research, analysis, synthesis, good judgment, perception, patience, and determination. The development of the vortal goal may take several dedicated, uninterrupted sessions by the project management team. It may also require field study, target audience surveys, or brainstorming sessions to arrive at a goal definition which is broad and yet specific enough to provide guidance for project development. The following steps will assist in the development of a vortal goal definition:

- Identify, list, and review the organizational mission statement, the specific applicable policies and doctrine that would impact the vortal and its content.
- Identify and categorize the vortal target audience(s).
- Identify user requirements of the vortal.
- Identify the general data, information, and services that the vortal will provide.
- Establish how the vortal shall provide the data, information, and services.
- Define the vortal goal.

4. Vortal Requirements

Vortal requirements mainly concern constraints in handling, managing and providing accessibility to vortal content: the data, information and services that are

identified within the vortal goal. Software requirements derive from the analysis of content and user interface requirements. Hardware requirements derive from analysis of handling, managing and accessibility requirements of the vortal content.

Content requirements define architectural needs of the vortal. Identification of specific data, information, and services provided within the vortal assists in development of content requirements by identifying what needs to be managed. Content should be limited to that which is necessary to meet the goals of the vortal. When trying to determine the content and what is important, (Collins, 2001) suggested the following guidelines:

- Identify roles within the organization that will use the content.
- Streamline information for these roles.
- Streamline information associated with the requirements derived from the vortal's goal.
- Continuously improve published vortal content based on feedback.
- Provide each targeted role with a perspective of available content.

The next significant step in content-requirements development is to identify the authorized operations that may be performed on the content. This is particularly important in the development of content information-assurance requirements. Identification of authorized operations assists in identifying content constraints as well as preliminary intrusion points and content weaknesses. It may be helpful to use universal modeling language (UML) "use cases" to assist in developing the requirements for the content. In addition, a version of use cases, called "abuse" cases, may help (McDermott and Fox, 1999). Furthermore, UML-based tools provide the source material for the development of vortal requirements and help provide source material for the vortal security policy.

Content correlation is particularly important for the mechanism to manage vortal content access and availability. The term refers to the correlation of target audiences to the various content groupings and subsequently correlating these groupings to allowable operations. A tool for this is a correlation matrix. Vortal content should not be changed without prior approval; a content approval process should be developed. This process

should evaluate content changes in light of security concerns as well as whether the proposed content supports the vortal goal.

The interface should follow consistent and acceptable standards of usability as supported by studies in human–computer interaction discipline. Interface requirements should be from the end-user’s perspective and should be based on a user-centered design which is grounded in the standards of usability and simplicity (Nielsen, 2000). If the interface does not meet the needs of the end user, the application will not receive the desired traffic or be used as designed.

Hardware requirements are also derived from analysis of management, handling, accessibility and availability requirements of vortal content as well as from the vortal goal.

5. Vortal Security Policy

The vortal security policy enforces the specific security requirements of the vortal. It covers protection for vortal content and the necessary vortal resources used to support vortal content management, handling, and processing. Formulating it necessitates studying the requirements for (1) protection from threats and vulnerabilities and (2) security services necessary to afford adequate protection based on content value and threats.

Content domains, which consolidate vortal content into groupings based on content sensitivity, help content security. The following steps are helpful:

- Identify the operational requirements of specific content domains.
- Identify vortal-availability requirements of these domains based on target audience needs.
- Define the priority of availability under reduced hardware capabilities, such as during maintenance and casualty response.
- Define the resource security requirements of each content domain in light of the organizational security policy.

Content availability requirements influence vortal architecture design. They require consideration of the following:

- Vortal Goal requirements
- Content security requirements

- System backup requirements
- Hardware and software maintenance requirements
- Normal operational requirements
- Organization policy (e.g. contingency planning and organization mission)
- Preliminary threat and vulnerability analyses of proposed vortal hardware and software.

Researching the known vulnerabilities of proposed vortal hardware and software can provide good material for the vulnerability analysis (Pfleeger, 1997). Defining possible access points on the network diagram as well as creating threat profiles using common criteria methodology can assist in the identification of threats to the overall vortal. In addition, as previously noted, abuse cases can develop system threats. Adequate software and hardware configuration management, discussed later, is also critical to maintaining vortal security.

It is important to ask what modes of vortal access will be allowed. For example, will the vortal be accessible through a wireless web browser, a corporate intranet or the Internet, and/or a dial-up connection? Modes of access should first be defined by the vortal goal and then by the needs of the target audience and current organizational policy. Different access modes may present different security considerations that will need to be addressed in the vortal security policy and the subsequently generated vortal security architecture.

Brainstorming or exploring ideas in a somewhat random fashion can suggest areas for security consideration. However, UML use cases provide a mechanism to quite thoroughly explore previously unconsidered issues (Larman, 1998). Once use cases are defined, a logical network diagram to graphically represent network architecture can greatly help rapid identification of network entry points and vortal access points.

A preliminary risk analysis provides information for the overall planning and development of the vortal (Pfleeger, 1997). This analysis can assess potential risks to the system prior to its rollout. It can also assist with contingency, safety, security, financial, and maintenance planning. Finally, a preliminary risk analysis provides the basis for a

comprehensive risk-management plan. This plan can assist in decision making and provide the prioritized actions necessary to appropriately manage risks.

C. ARCHITECTURE DEVELOPMENT

Architectural design is necessary for systems that are standardized, productive, secure, and defensible. It provides a guide map to maintain focus in the development of a system so that it satisfies its intended goals and objectives. Further, architecture helps to solidify a development route and methods used. It increases credibility of the development process and provides a framework where measurements can be obtained, in a standardized way, to prove or validate the system.

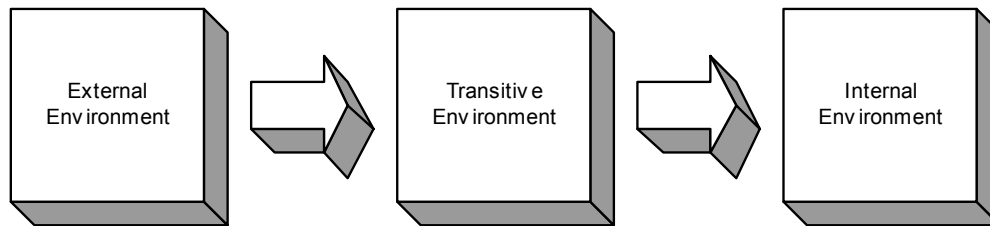


Figure 25. High Level ORIS Architectural Environment

1. Organizational Information System (ORIS) Architecture

ORIS architecture is a high-level conceptual architecture derived from organizational requirements. Requirements are translated into one or more processes, which may be illustrated as abstract modules displaying the main conceptual areas of the architecture. Figure 25 illustrates a simple example of ORIS architecture. Each module can be broken down into standardized non-specific components, which further refine the abstract diagram (e.g. terminal elements, and network elements). Figure 26 further refines the ORIS architecture described in Figure 25.

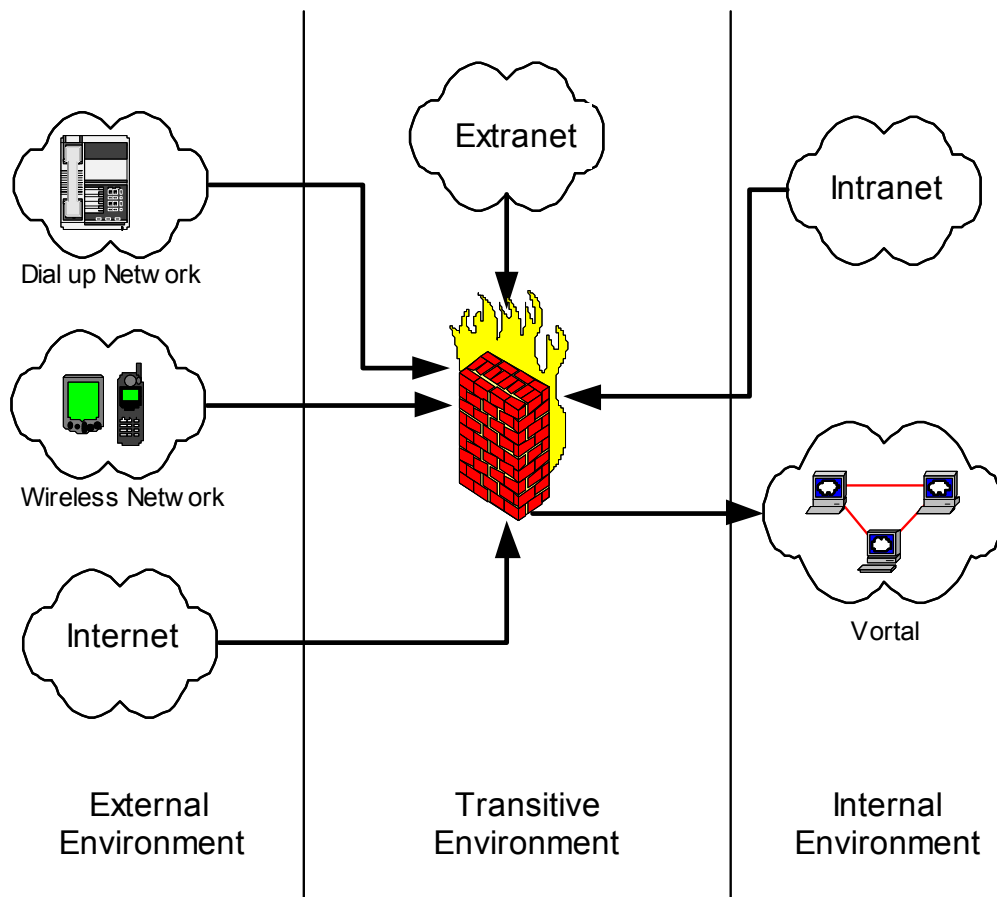


Figure 26. High Level ORIS Architecture Refined

A high-level ORIS architecture can define, in a simple way, the overall ORIS requirements. In addition, it may be used to help develop boundary limitations for the subordinate ORIS architectural entities that represent significant processes and projects within the organization. When thoroughly defined, ORIS architecture can provide a standardized set of rules for management, handling, and a common terminology for many different information systems. Finally, it provides a reference mechanism which supports the rapid development of subparts by defining the context for integration and the limiting boundaries.

The ORIS security architecture is defined within the constraints of organizational requirements and derived from the ORIS security policy as well as the larger ORIS architecture. Figure 27 refines, within the context of security, one portion of the ORIS architecture example provided in Figure 26 based upon communications protocols as well as physical and administrative environments.

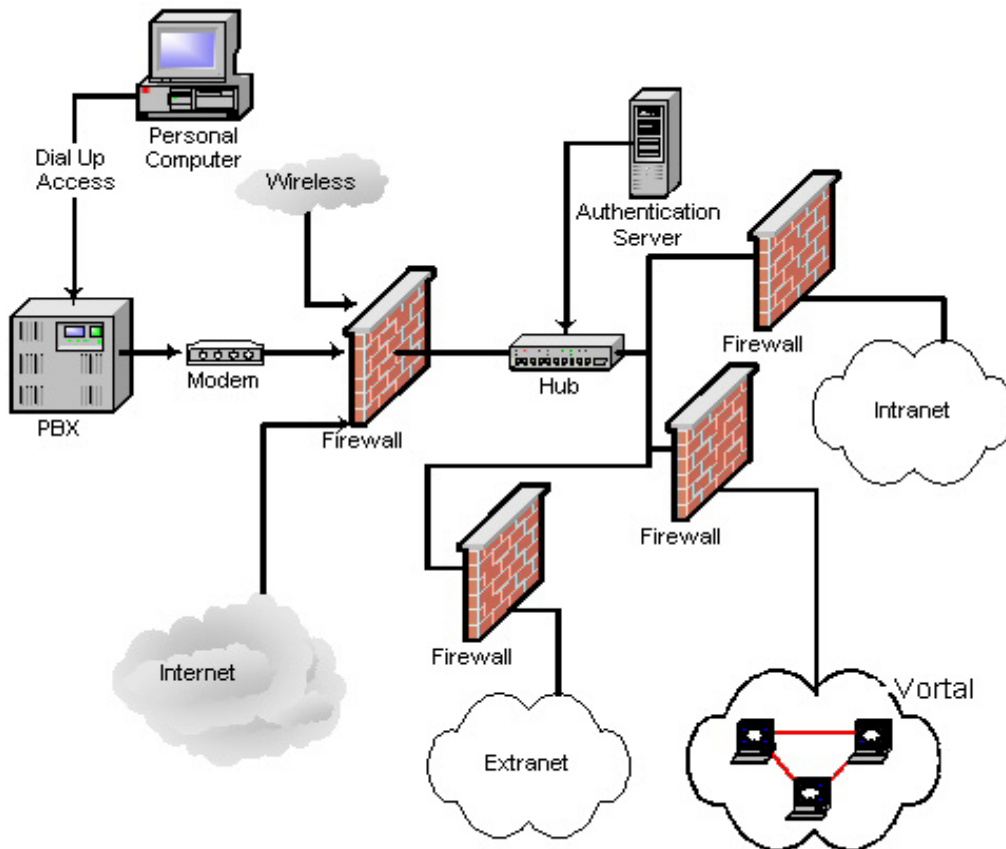


Figure 27. Example ORIS Security Architecture—Dial Up Network

2. Vortal Architecture

Vortal architecture is a necessary part of the development process. It provides a guide map to the implementation of the vortal. It provides a framework which sets limiting boundaries and focuses the efforts of vortal development.

For ease of management and handling as well as implementation, content should be subdivided into content domains. These domains contain categorized groupings of data, information, and services, but also authorized operations on the content. For ease of management and handling, each content domain should be further subdivided by data-object storage method (will the data be stored in a database, a data file, or an application data file?) This further subdivision provides clarification and the means to create a hardware design that will ensure higher levels of data security.

Vortal requirements help to determine what features are required for the core management system. A decision whether to design it or use a prepackaged configurable software application should be made based on vortal requirements, content domain requirements, and content value plus other requirements from entities such as higher authorities, the vortal security policy, or applicable doctrine. Any determination for specific technology such as the extensible mark-up language (XML) or some other standard must be evaluated similarly.

Proper software application design and configuration reduces security problems. When applications are correctly designed, they handle proper improper input data gracefully and do not provide unauthorized root access to the application server, perform unauthorized operations, or gain unauthorized access to resources. The application should be tested over a wide range of data inputs including null inputs and incorrect values. To assist with vulnerability reduction, the design can designate another protection layer.

Data storage and handling strategy should have provisions for rapid recovery in the event of disaster as well as mechanisms to ensure confidentiality, integrity and availability of data. It should provide security limitations compliant with the vortal security policy as well as the ORIS security policy. User-level access to data would be limited to indirect access via trusted vortal accessible software applications.

3. Vortal Security Architecture

Vortal security architecture is a logical architecture that details design rules as well as all system aspects related to security (Gasser, 1988). There are four areas of concern.

Vortal components are composed of the objects (e.g. hardware, software, and transmission systems) that comprise the logical design of the vortal architecture; these components that may be used as initial starting points for subsequent security analyses.

Identification of threats, vulnerabilities, and risks help to indicate the weaknesses and associated costs within the vortal. They help to provide justification and direction to the development of security. They may be prioritized for the concentration of security resources.

Security-policy critical application points are logical decision points that directly support the vortal security policy. It is important that the vortal does not have too many or too few. Too many points mean that the vortal has too many potential weaknesses; too few points mean that the vortal may not be practical to use. Therefore, a balance must be obtained.

Security services provide the essential security support to vortal security-policy critical application points. Each service must be designed so that authorized transactions are allowed to occur while unauthorized transactions are not. There are six main security services (International Standards Office, 1998):

- Authentication: This establishes proof of user identity and provides subsequent authorizations or rights for the identified user.
- Confidentiality: This assigns the authorized user their material disclosure limits.
- Integrity: This assigns to the authorized user any limits to their ability to make changes to data.
- Access control: This determines user authority. Subsequently, it provides to the controlling mechanism the explicit use of computer resources based on the user's authority.
- Non-repudiation: This provides evidence to prevent unilateral modification or termination of obligations in an authorized computer transaction.
- Availability: This provides automated routing of information, load balancing, and system traffic accommodation to ensure that resources are usable on demand.

4. Design Assessment

Design assessment is the process of validating the vortal architecture and vortal security architecture to ensure that the implementation correctly met the intended goals of the vortal. The following are recommended steps.

- Define the scope of the assessment.
- Define the metrics to be used within the assessment to establish what is to be measured and the limits of the measurement.
- Establish the method of measurement. This is important because it determines the viability of measurement results. This is important when determining trends and recommending changes based on those trends.

- Establish the measurement collection mechanism. This provides the precision and determines the accuracy of the metrics used. It should be tested before it is used to ensure that the measurement taker is adequate collect the data.
- Establish how often the data should be collected before the collection of data begins. It should be determined from the metrics used and the intended use of the data.
- Sort and interpret the data to determine any trends and to formulate appropriate feedback. A variety of interpretation techniques should be used.
- Provide feedback to the data collection mechanism, method, and metrics.
- Provide goal feedback based on trends in the collected data.
- Provide architecture feedback based on trends in the collected data or obvious deficiencies in the vortal architecture or vortal security architecture. Submit it for change approval prior to implementation.

D. IMPLEMENTATION AND PROTOTYPING

Significant benefits of a well-planned solution are lost if a solution is poorly executed. For example, if a vortal solution is well designed and properly fielded but incorrectly configured then the designed security mechanisms may be overcome or bypassed. This may result in compromise of not only the local area network that may be connected to the vortal but also the data, information, and other services that are available to the vortal. But similarly, a poorly planned solution that is executed well cannot reasonably guarantee any degree of security within an information system. For example, if a vortal architecture requires both data and applications on the same server and a user gains unauthorized access to the server because of a buffer overflow within a vortal application, the user gain complete access rights to the data to which they may not have authorization.

One of the many tools for implementation of a vortal project is small-scale prototyping. It can be cost-effective if it creates opportunities to identify and correct rough spots before the full-scale implementation is undertaken. In addition, proper prototyping can validate or invalidate a particular solution and provides a mechanism from which to derive lessons that can be applied to the improvement of the full version of the vortal.

1. Designing the Implementation Plan

A good way to implement a vortal is to start with a plan or implementation strategy. If the vortal will be a significant service for the organization, then it will be a long-term endeavor. Successful long-term endeavors require a well-developed plan for implementation. But to develop and implement a vortal there is no single “cookbook” approach to making a plan.

A vortal can cause changes in how organization work is accomplished. Implementation planning should identify the sequence to make those changes and the people involved. A well-conceived plan provides a mechanism to avoid the “false starts” typical of unplanned initiatives. An approved plan demonstrates leadership commitment and helps organization members understand their role in leading and supporting the vortal project. Senior-level commitment is often deepened through inclusion or participation in the implementation planning process.

- A good implementation plan with milestones will assist an organization to:
- Set up a timeline for actions.
- Allow for assessment of progress on the vortal.
- Establish necessary mechanisms to cope with the changes associated with integrating the vortal into the organization's way of doing business.
- Begin focusing decision-making processes towards vortal support.
- Create and maintain a user-centered focus regarding the vortal.
- Understand better how organization work is accomplished.
- Improve leadership thinking.

The following are some cautionary notes to consider when developing the vortal implementation plan:

- Don't *front-load* the implementation plan. Often organizations become too optimistic about the quantity of work and assign early milestones, yet are reticent to assign additional resources to complete the rest of the work.
- Perform a logic check on an implementation plan to ensure that tasks are sequenced correctly with adequate resource allocation.
- Implementation plans can be modified when the need arises. But too many due dates postponed may signify a lack of proper planning and commitment to accomplish the project.

- An effective way to ensure that the end goal is met is to display and promote the plan as much as possible throughout the organization. This can be accomplished by providing periodic updates to organization members.
- Periodically updated copies of the implementation plan should be provided to the personnel who are assigned milestones to accomplish. This reminds them of their tasks as well as indicates the impact on the overall project if they fail.
- Use software and hardware configuration management. An improperly configured system provides many avenues for unauthorized system access.
- Include only necessary hardware and software functionality to meet the requirements of the architecture.
- Consider the logical network diagram as a tool from which to draw information in developing the implementation plan.

2. Prototyping the Design

Prototypes are a means of starting the vortal implementation process and a tool for improving the quality of the delivered vortal. They are small design assessment steps to help direct the development process. Development efforts will always yield mistakes, but mistakes can be minimized through the use and analysis of prototypes. Prototypes demonstrate specific objectives which support the vortal goal and demonstrate functionality in light of security. In addition, they help maintain a user-centered focus throughout the vortal project.

The vortal development team and resource managers decide what will be prototyped. One method is to take the vortal logical diagram and identify a part whose performance is measurable, has a high probability for success, and has a high project impact. Another method is to develop a broader prototype within a content domain. With prototypes, the vortal development team creates opportunities to apply what they have learned during design.

Prototypes can provide useful information. Limited scope prototypes with clearly defined objectives seem to provide more data than broad-scope prototypes with vague objectives. Data obtained from prototype evaluation can be used to improve not only the vortal but the methods for building a prototype.

E. TESTING, VALIDATION, EVALUATION, CERTIFICATION AND ACCREDITATION

Testing the full implementation will identify flaws in the functionality and performance of component modules within the overall system. In addition, testing can be applied to the overall system design as well. For example, testing can identify flaws in the hardware, software, and interface usability. But successfully passing a test does not demonstrate the absence of a flaw. In addition, it is difficult to achieve adequate test coverage because of the complexity of various inputs and states of a system. Furthermore, powerful “white-box” or internal-structure testing requires modification of the implementation to obtain data, which affects the solution and may become a subsequent source of vulnerabilities (Pfleeger, 1997).

Validation and evaluation are used to prove system correctness. These efforts can be used to convince users that a system correctly implements stated goals. In addition, it can prove that a system correctly implements stated policies. Validation is less rigorous than reducing a system to a theorem and proving it. Validation is more general and attempts to convince people of the correctness of a solution. Validation may be performed several ways:

- Requirements testing may be used to cross-check system requirements with execution-time system behavior. The goal is to demonstrate that the system performs all functional requirements. But often this only demonstrates that the system does everything required in one situation, not that the system doesn’t do what it should not do.
- Design and code reviews rely on system designers and programmers to scrutinize system design and system code. The goal is to identify and correct any incorrect assumptions, errors, inconsistent behavior, or faulty logic during system creation. Success depends on the rigor of review.
- Module and system testing uses selected data to check system correctness. It relies on programmers, system designers, or independent testing teams to conduct these checks. Test data is organized to examine execution paths, conditional statements, output reports, variable changes, etc. The goal is to check all objects methodically.

Most users are not security experts. Therefore, evaluation by an independent third party can be desirable to prove adequacy and accuracy of test coverage, validate correctness conclusions, or determine that a system correctly implements security policy.

Independent experts can “review the requirements, design, implementation, and assurance evidence of a system.” There are many evaluation schemes available. Recently Common Criteria seems to be gaining momentum as a scheme for evaluation.

Certifications determine if technical mechanisms in the vortal effectively meet a particular standard, e.g. FIPS 140 or NSA Type 1 encryption. This can be useful for insurance and liability analysis as well as for the development of risks for system level risk management. Certification is defined as the technical evaluation of security features and other safeguards (Department of Defense 2001). It is used to support the accreditation process and can be used to define a minimum standard for security. When changes are made in the vortal system, recertification of the changed system is necessary.

Accreditation is endorsement by an accrediting organization that a vortal’s performance successfully meets criteria. Accreditation criteria can assist in the development of a risk management plan as the criteria provide a reference from which to define areas for further investigation. In addition, the act of gaining accreditation can be important in building trust with users as well as with organizations insuring against liabilities and lapses in performance. Accreditation is defined as the authorization by a designated approving authority that an information system may be placed into operation (Department of Defense, 2001). It confirms acceptance of a system’s configuration, design, and architecture.

F. FEEDBACK

Sometimes referred to as maintenance, the feedback process provides the mechanism by which improvements are made to a working system. An effective and efficient vortal system is the result of abundant feedback. Improvements cannot be made if deficiencies are not openly identified and addressed in an honest and forthright manner. A well-defined, dedicated, and robust feedback process has significant benefits: It can assist in fielding timely improvements to the vortal system, and can produce consistent feedback that is both thoughtful and usable.

THIS PAGE INTENTIONALLY LEFT BLANK

V. IMPLEMENTATION

To illustrate concepts and methodology described in the last chapter, a simple Navy development example of a secure vortal architecture and associated security architecture is presented. The example focuses on architecture development and illustrates required efforts necessary to produce a secure vortal, but does not include an implementation plan.

The example vortal supports a fictitious ship, USS NEVERSAIL; it is a typical naval ship, roughly a cruiser. Assume the normal crew size is about 250 personnel with 25 officers (a commanding officer, an executive officer, department heads, and division officers). Most ships of this type carry land attack missiles, ship-to-ship missiles, surface-to-air missiles, anti-submarine weapons, and a main gun system for naval surface fire support. Normal warfare areas assigned to this ship could include undersea warfare (ship or helicopter to submarine engagements), strike warfare (land attack using cruise missiles), surface warfare (ship-to-ship engagements), air warfare (surface-to-air engagements), theater ballistic missile defense, maritime interdiction operations and leadership interdiction operations. The majority of the shipboard computing environment is made up of multiple redundant weapons computing systems that are standalone and receive their inputs directly from onboard sensors and sometimes from other Battle Group asset sensors. The remaining shipboard computing environment is handled through other IP based computing networks. These networks exist to handle the organizational administrative support requirements as well as to automate some manual processes within shipboard command and control.

Command and control is currently handled through the abovementioned weapons computing systems, through manually generated human input (e.g. radio messaging system, radio telephone, signal flags, semaphore coded messages, flashing light messages), or through these other IP based computing networks. The IP based computing systems that include the network supporting this example vortal have become the popular systems for timely collaboration, integration, aggregation and consolidation of vital data, information, and other services that are associated with command and

control as well as administrative support. The USS NEVERSAIL vortal example provides mechanisms to support information superiority through timely knowledge management. For a ship, information superiority is the foundation of successful command and control.

A. ASSEMBLE THE RIGHT TEAM

Traditionally, IP based networks are designed, tested, installed, and owned by Navy Network Warfare Command via SPAWAR Systems Command. Shipboard personnel are assigned to perform network administration and some rudimentary network maintenance. Maintenance of these networks is funded through an assigned ship maintenance budget. Frequently, resource sponsors that are external to the ship will provide funding for initiatives like a shipboard vortal that has potential to enhance the warfighting capability of military assets.

The exact makeup of a design team will depend on the knowledge and skills of constituent members as well as the vortal goal. Where shortfalls in knowledge or skill exist, it may be of benefit to hire contract experts to cover those discrepancies. Generally, for the design and planning of the vortal, a team should be formed that includes knowledgeable members of the vortal target audience, supervisors of the personnel assigned to perform network administration and maintenance, resource sponsors both external and internal to the ship, any required contractors, and representatives of the organizations that support the greater network architecture (e.g. Navy Communications and Telecommunications Stations, Navy Network Operations Centers, etc.).

Implementation constraints are defined by the limitations of the communications network architecture to be used (e.g. the shipboard system-high classified network architecture, satellite communications network architecture, network operations center's system-high classified network architecture, etc.) as well as network accreditation, certification and operations standards (e.g. IT21, DITSCAP, TF WEB architecture standards, DoD website requirements). These standards and requirements will provide the necessary guidance for software and hardware to help design and plan the vortal. Contract requirements and limitations are outlined in applicable Navy supply regulations

and publications. In addition, it may be beneficial to research existing contracts and identify if any existing contract can be modified to accommodate the contract requirements for the vortal. This will considerably reduce the amount of effort to administratively process a contract requirement.

B. CONCEPT & CONTENT DEVELOPMENT

1. Organizational Policies and Mission

Assume a review of USS NEVERSAIL's organizational policies and mission statement yielded a list of guidance and capabilities, selected by the USS NEVERSAIL's chain-of-command for vortal inclusion based on their impact on the organizational mission:

- Provide daily status of operational readiness and training.
- Provide a mechanism to manage the system of processes for material maintenance management of ship equipment.
- Maintain communications with higher authority and other battle group and expeditionary strike group assets.
- Safely navigate the ship.
- Provide up-to-date access to publications and equipment technical manuals.

2. User Inputs

User inputs were obtained from surveys and interviews. Inputs were grouped and categorized according to similarities. Suppose the USS NEVERSAIL's chain of command identified the following as the highest-priority user requirements:

- A more user-friendly and available message processing system. These messages include: situation reports, personnel administrative messages, guidance from flag officers (e.g. admirals, generals, etc), contact reports (with other ships, submarines, or aircraft), ship operational tasking orders, ship administrative tasking orders (e.g. ship tour for a visiting dignitary, assignment as acting squadron commander in the absence of the ship's squadron commander) ship support requests (e.g. equipment part requests, import shore services, tug support for getting underway or returning to port, etc) as well as communications of a general nature to support a tasked operation.
- Weather information for safe navigation. This includes information regarding weather in advance of a ship's movement as well as weather information that may have an impact to the ship during a tasked operation.

- Parts ordering and a material maintenance trouble-call system.

3. Vortal Target Audience

Suppose from brainstorming and categorization, a list of vortal target-audience groups was produced covering the significant stakeholders:

- Higher authority
- Ship's company
- Shore-based support organizations
- Other Battle Group or Expeditionary Strike Group assets

4. Vortal Content

Suppose a vortal-project team-brainstorming session yielded the listing of data, information and services that need to be provided, as shown in Table 4.

| Resource | Requirement | Audience |
|--|---|---|
| Equipment casualty reporting information | Provide daily status of operational readiness | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Training status information | Provide status of training | Higher authority, other battle group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Fuel and weapon status information | Provide status of operational readiness | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Troublecall information | Mechanism to manage material maintenance system | Ship's company, shore-based support organizations |
| Parts ordering | Mechanism to manage material maintenance system | Ship's company, shore-based support organizations |
| Weather information | Safe navigation of ship | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Publications and equipment technical manuals | Provide access to up-to-date publications and equipment technical manuals | Ship's company, shore-based support organizations |
| Email messaging system | More user-friendly and available message processing system; communicate from higher authority, other Battle Group and Expeditionary Strike Group assets | Ship's company |
| Chat | Communicate with higher authority, other Battle Group and Expeditionary Strike Group assets | Ship's company watchstanders, specific Battle Group and Expeditionary Strike Group watchstanders |

Table 4 Correlation Table For Combined Resource and Target Audience to Requirements

C. REQUIREMENTS DEVELOPMENT

1. Organizational Requirements

In the case of USS NEVERSAIL, suppose the organization is required to maintain two system-high networks using identical network architectures and one multi-level security network. Of the two system-high networks, one is classified and the other is unclassified. Due to space and electrical power limitations, USS NEVERSAIL system redundancy is limited to a single set of backup servers per network. If the functionality is

vital, additional server installations can be provided. “Reachback” connectivity to the terrestrial portion of each network is via a single 14 megabits-per-second satellite channel shared by the three networks; satellite bandwidth can be dynamically allocated on the ship. Network traffic prior to vortal implementation through the satellite channel for all networks is no greater than 6 megabits per second, so design must find a way to increase it.

Users access networks through physically connected network workstations. Secure wireless connectivity on the ship is being considered for the system-high unclassified network. Each system-high network is router-based with a fiber optic backbone. The multi-level security network is switch-based with all connectivity through dedicated fiber optic cable.

2. Organizational Security Policy

Suppose for the USS NEVERSAIL, generalized preliminary evaluations of the three networks indicate that the following network resources require protection: workstations, network devices, network servers, cryptographic equipment, cryptographic equipment key material, and satellite equipment. The following policies exist:

- Physical security policies that govern the physical security requirements for network resources are implemented.
- Password policies that govern selection and handling of passwords are implemented. User passwords are required to be 10 alphanumeric and special characters. System Administrators and network support staff passwords are required to be 14 alphanumeric and special characters. In addition, passwords are effective for no greater than 90 days. Password-cracking software is used to audit user passwords.
- System-administrator policies dictate documentation and general administrative requirements as well as allowable operations, actions, accountability, and limitations on system administrators. Security audits are conducted daily at the beginning of each network administrator shift. Current network operations and staffing allow for three eight-hour network administrator shifts per day.
- Network backup policies dictate requirements for each shift as well as each computing day.
- Network device configuration policies dictate the required actions, audits (successful and failed user attempts), documentation, and limitations for authorized network support staff.

- Network server configuration policies dictate the required actions, requirements, documentation, and limitations for authorized network support staff that setup and configure network servers.
- Workstation configuration and maintenance policies dictate the required actions, documentation, audits, and limitations for network-support staff that setup, configure, and maintain network workstations.
- Network-resource user policies are implemented and dictate the responsibility, accountability, and limits of network resource use for authorized network users.
- Disaster recovery policies detail requirements, responsibilities, accountability, documentation, and actions of applicable personnel.
- Intrusion-detection and anti-virus software are installed and properly configured. In addition, machine-address code filtering is implemented in all network devices.
- Training policies dictate the frequency, quantity, and type of training required for network administrators, network support staff, and users. Training includes scenario drills, classroom knowledge, and individual supervised hands-on training. Training program audits are conducted monthly by the USS NEVERSAIL's chain of command.
- Individual policies are reviewed annually by all network administrators, network support staff and support staff for clarity, revision and familiarity.

3. Vortal Goal Definition

Suppose that through a series of meetings, the NEVERSAIL chain of command developed the vortal goal statement from the list of desired capabilities, target audience stakeholders, and user inputs:

The NEVERSAIL vortal is representative of USS NEVERSAIL and provides operational support by access to up-to-date status information about operational readiness, training information, publications and equipment technical manuals, weather information, and material-maintenance management and communications to ship's company, higher authority, shore-based organizations and other Battle Group or Expeditionary Strike Group assets. We provide this support through Intranet accessibility over a dedicated system-high classified network.

4. Vortal Requirements

In the context of the USS NEVERSAIL, several factors were considered in formulating requirements: subject content, information classification, target-audience network-access availability (i.e. number of workstations available), allocated satellite

bandwidth and throughput, vortal bandwidth and throughput requirements, and organizational information system architecture. These factors were prioritized. The highest priorities were content-information classification and the availability to the target audience. Based on these priorities, it was decided to implement the vortal on the system-high classified network.

a. Content Requirements

Figure 28 shows the “basic” hybrid information environment “use case” derived from the MODELS Information Architecture action sequence.

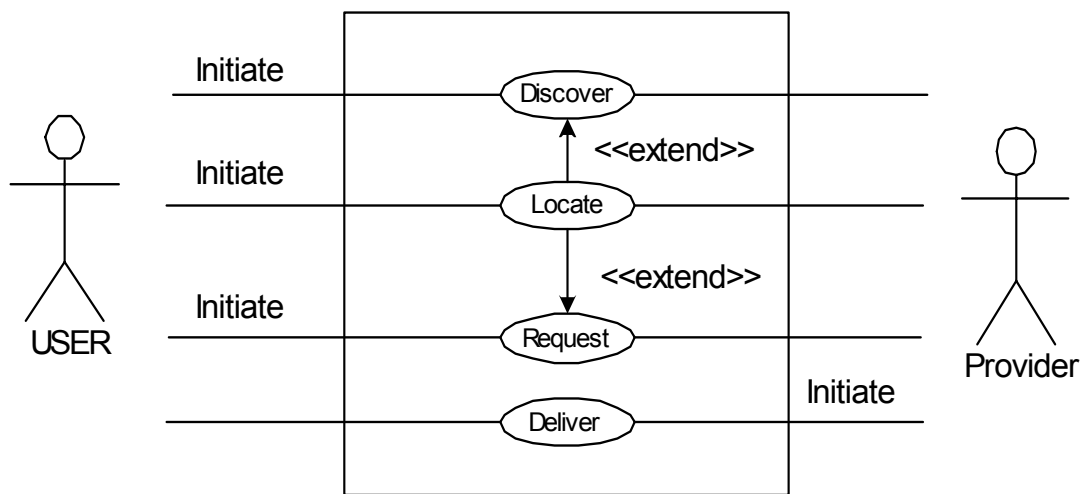


Figure 28. “Basic” Discover-Locate-Request-Deliver Use Case (From Gardner, 1999a)

The “<<extend>>” notation suggests the performance of one action is an add-on to another. Initiation of “Locate” may be performed by either the user or system during the discovery phase where other resource-description information and locations are shown. Another allowed possibility is that a resource may be requested without a selected location. The system would then find possible locations and offer a choice of them to the user or selecting one based on user preferences (Gardner, 1999a).

Following Gardner, the basic use case would support the following:

- Cases in which equipment-part descriptions and location information have separate providers. Users may first make work-level resource discovery. Subsequently, they may locate desired resources while moving to the

instance level. This is likely when ships and shore-based storage facilities possess quantities of the same part in their storerooms.

- Cases in which equipment-part descriptions and location information have the same provider. For example an equipment part may be located at various “mirror” storerooms in USS NEVERSAIL.
- Cases in which requests are initiated after discovery. Then during item requests a location must be selected. For example, a user may request weather information and be offered a choice of mirror sites from which to download this information.
- Cases where no information-discovery phase is required. For example, users who want to obtain weather information are directed to the nearest mirrored site.

It is expected that a hybrid information environment will mix information, behaviors and possibilities for interoperability. An important concern is that sufficient information regarding a resource must exist prior to delivery. In addition, delivery could take place outside system boundaries. For example, it is expected that a Fleet Industrial Supply Center (provider) would initiate a delivery task within the electronic system, for logging and tracking purposes, even if actual equipment delivery is physical and not electronic. In other cases the delivery could occur through the electronic system.

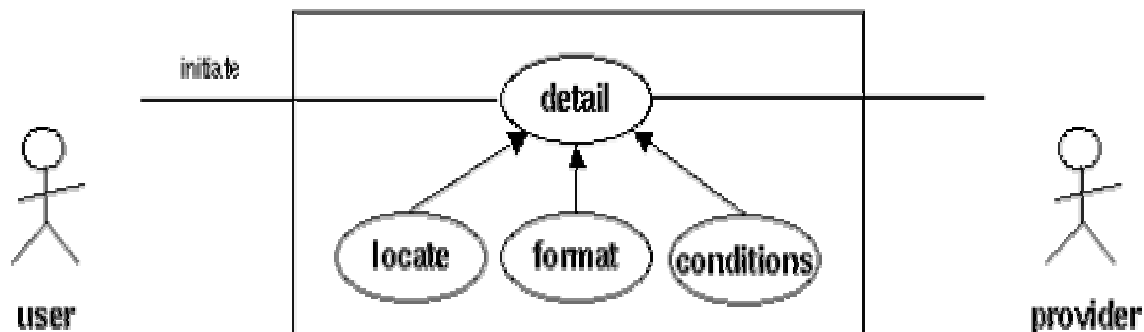


Figure 29. “Detail” Use Case Describing the Discovery Level of Resources (From Gardner, 1999a)

Multiple editions, formats, languages, or versions of information with similar content may be available at different locations. This can create difficulties within queries if different levels of detail are required depending on context. Figure 29 illustrates the “detail” use case describing the functional-discovery level of resources. Previously the treatment of the “locate” was as a separately-driven user action. This

works well in a hybrid information environment where a single query is initiated to multiple providers for a specific resource. In other circumstances, location may be only one detail required to specify a deliverable resource. To support the acquisition of such descriptors as versions, formats, and other details, the “detail” use case is employed to ensure that the correct ‘instance’ of a deliverable resource is captured.

“Detail” use cases provide a selection of details to narrow a particular resource’s instances. Arrow notation in Figure 29 identifies locate, format, and conditions as specific use-case variants. These use cases may be viewed as a first step in discovery because they provide a listing of resources (e.g. the same document in formats RTF, HTML, and DOC). An important detail is the conditions and terms related to a resource, as a cost that varies across providers. For example, a user can view a document (resource instance) on a remote server (provider) with an initial load refresh of 15 seconds, or the document can be downloaded over the network at a cost of 15 minutes.

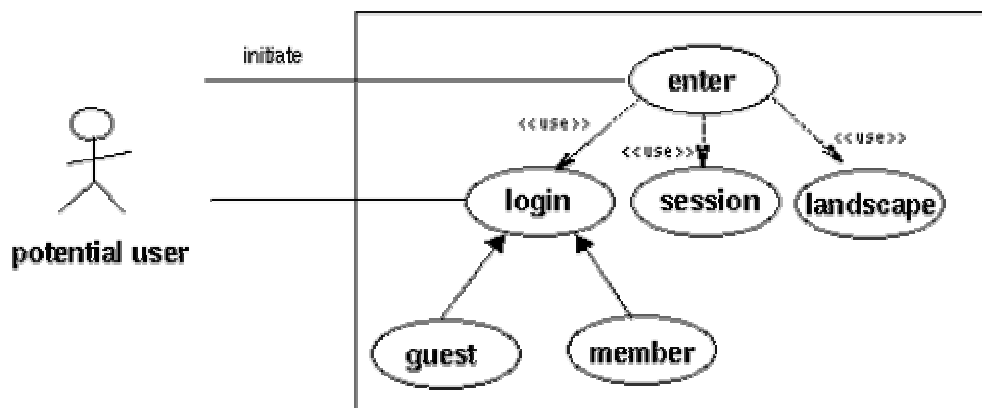


Figure 30. “Enter” Use Case for System-Entry Behavior (From Gardner, 1999a)

The “enter” use case for user input (Figure 30) supports tailored information landscapes with authenticated user access to provide the appropriate user authorizations, preferences, and entitlements. In addition, in hybrid information environment it is typical for current session information, such as results from the most recent search to be maintained. There may be a need for authenticated access for a guest user in addition to a member user. Their tailored information landscape is limited to what is available in the current session as their profile, whereas user preferences via profiles are available to member users. Note that if a user is not entitled to see it then an object

will not be made visible. Table 5 displays several such variations on the “enter” use case scenario.

| | Normal | Variation 1 | Variation 2 | Variation 3 |
|---------------|--|---|---|--|
| Step 1 | User loads top-level Web page. | User attempts to access a service requiring authentication. | User loads top-level Web page. | User loads top-level Web page. |
| Step 2 | User submits user-name and password. | User attempts guest access. | User submits user-name and password. | User submits user-name and password. |
| Step 3 | New session initiated and information landscape presented based on user profile. | After successful authentication, user sent to service they attempted to access. | System continues a previous session and displays its information landscape. | Authentication fails; return user to step 2 for another attempt. |

Table 5 “Enter” Use Case Scenario Under Normal Conditions and With Several Variations (Adapted From Gardner, 1999a)

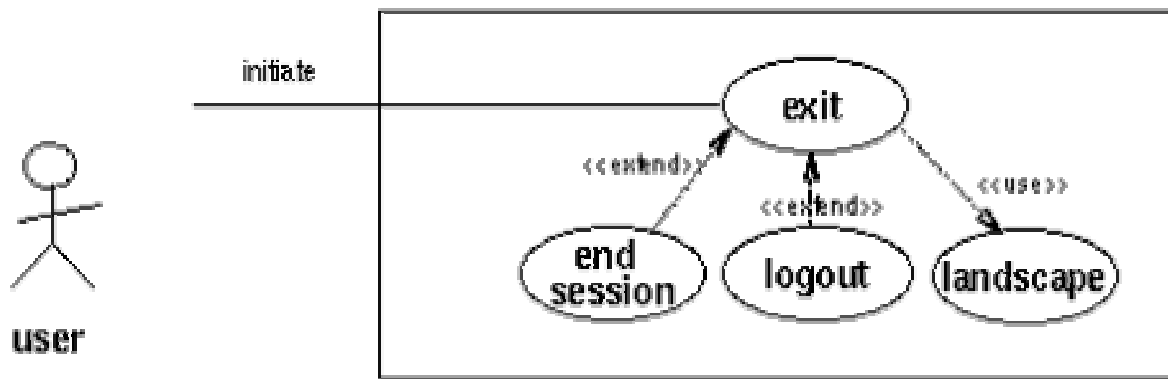


Figure 31. “Exit” Use Case Describing System Exit Behaviors (From Gardner, 1999a)

During system exit (Figure 31), users could be offered an unambiguous fixed exit that clears the session or users may have alternatives such as a predefined time-out period or an “end session” use case.

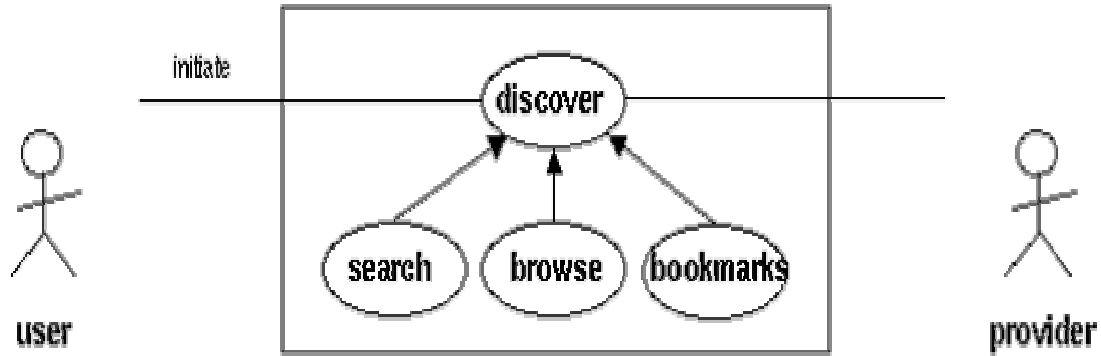


Figure 32. “Discover” Use Case Describing Resource Discovery Behaviors (From Gardner, 1999a)

The “discover” use case uses ‘tools’ to crawl through the information landscape. This can include free-text and metadata search as well as browsing, viewing bookmarks and what’s new, and so on. Table 6 shows several variations.

| | Normal | Variation 1 | Variation 2 |
|---------------|--|--|---|
| Step 1 | Discovery tool chosen by user. | System makes tool available without user selection (e.g. free-text search box on every page). | Discovery tool chosen by user. |
| Step 2 | Required configuration parameters entered by user. | Tool configured in context of current information landscape, user profile, and/or session information. | User supplies a set of requested parameters. |
| Step 3 | Service providers deliver results. | Search refinement tool offered to further refine a listing of current search results. | Negative response tool is used if no appropriate results are found. |
| Step 4 | Results displayed to user. | Results displayed to user. | Results displayed to user. |

Table 6 “Discover” Use Case Scenario Under Normal Conditions and With Several Variations (Adapted From Gardner, 1999a)

The use cases provided are only a few of the many that could be developed for the USS NEVERSAIL vortal. In addition, “abuse” cases can identify unacceptable behavior. An abuse case is defined as a “specification of a type of complete interaction between a system and one or more actors, where the results of the interaction are harmful to the system, one of the actors, or one of the stakeholders in the system

(McDermott and Fox, 1999).” An abuse case focuses on transactions between actors and the system that result in harm. They describe the actors, their resources, skills and long term goals.

For example, consider the abuse case of a *Script Kiddie* on the system-high classified network. They typically work alone, although they may share information they obtain with other Script Kiddies. They have limited technical expertise and perform most of their activities using tools and techniques developed by others. They desire to demonstrate their technical ability and may have unlawful objectives including theft and vandalism. They have available hardware, software, and network access. Script Kiddie abuse may be:

- Installation of unauthorized software utilities using administrative or user privileges on a host computer.
- Single-instance control of an administrative account or session on a host computer.
- Single-instance control of a privileged user account or session on a host computer.

An example abuse case would be when the Script Kiddie initiates or requests a session on a network host using the TCP/IP protocol and an unauthorized software utility. A session is established; if the session is with sufficient privilege or a system software flaw is exploited such as a buffer overflow, the Script Kiddie will attempt to initiate an operating system level support function such as a command shell, file manager, editor, or debugger for the purpose of loading additional unauthorized software to obtain increased privileges or to copy privileged files. If any of these privileged files increase Script Kiddie abilities or meets their interests and objectives, the files are downloaded and copied for future use. If vandalism is their objective, a Script Kiddie may attempt to alter the privileged files or replace it with other files.

b. Content Correlation

Content correlation connects allowable object operations with target audience groupings and objects. Table 7 shows the correlation matrix for the USS NEVERSAIL secure vortal project. Table 8 shows the assigned user roles for the vortal.

| Subjects Objects | Content Managers | Content Maintainers | Browsing Users |
|----------------------------------|---|---|---|
| Links | Read Only: Can see the link Read/Write: Can see the link and modify any of its properties | Read Only: Can see the link Read/Write: Can modify the link's name and description | Read Only: Can see the link Read/Write: Can see the link |
| Folders | Read Only: Can browse the folder Read/Write: Can submit, move and remove links, move, delete, or modify the folder Read/Write/Approve: Can add and remove links, and approve links submitted byh users or imported by agents | Read Only: Can browse the folder Read/Write: Can submit, move and remove links, move or delete folder Read/Write/Approve: Can add and remove links, and approve links submitted by users or imported by agents | Read: Can browse the folder Read/Write: Can submit links to the folder, pending approval Read/Write/Approve: Can submit pre-approved links into the folder |
| Publications | Read Only: Can see the publication Read/Write: Can modify or delete the publication and approve its issues | Read Only: Can subscribe to the publication over the Web Read/Write: Can approve issues of the publication | Read Only: Can subscribe to the publication over the Web Read/Write: Can subscribe to the publication over the Web |
| Interoperability Modules | Read Only: Can use the Module and personalize it where applicable Read/Write: Can modify or delete the Module | Read Only: Can use the Module, and personalize it where applicable Read/Write: Can use the Module, and personalize it where applicable | Read Only: Can use the Module, and personalize it where it applicable Read/Write: Can use the Module and personalize it where applicable |
| Data Sources | Read Only: Can see the and crawl the data source, and create links to information in the data source Read/Write: Can modify or delete the data source | Read Only: Can submit links to information in the data source | Read Only: Can submit links to information in the data source |
| Properties | Read Only: Can see the property's values Read/Write: Can modify the property's values or delete it | Read Only: Can see the property's values | Read Only: Can see the property's values |
| User Groups | Read Only: Can see the user group Read/Write: Can add and remove members | N/A | N/A |
| Autonomous Crawling Agent | Read Only: Can see the agent Read/Write: Can modify and delete the agent | N/A | N/A |
| Document Types | Read Only: Can see the document type Read/Write: Can modify or delete the document type | N/A | N/A |
| Jobs | Read Only: Can see the job Read/Write: Can add or remove operations, modify the job's schedule, or delete the job. | N/A | N/A |

Table 7 Default User Roles and Privileges to Objects (Adapted From Plumtree, 2000)

| Resource | Content Manager | Content Maintainer | Browsing User |
|--|--------------------------------|---|---|
| Equipment Casualty Reporting Information | USS NEVERSAIL Operations Dept | USS NEVERSAIL Operations Dept | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Training Status Information | USS NEVERSAIL Operations Dept | USS NEVERSAIL Operations Dept | Higher authority, other battle group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Fuel and Weapon Status Information | USS NEVERSAIL Operations Depts | USS NEVERSAIL Engineering and Operations Depts | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Troublecall Information | USS NEVERSAIL Engineering Dept | USS NEVERSAIL Engineering Dept, Port Engineer, Tender Repair Officer, SIMA Project Officer, or Shipyard Project Officer | Ship's company, shore-based support organizations |
| Parts Ordering | USS NEVERSAIL Supply Dept. | USS NEVERSAIL Supply Dept. | Ship's company, shore-based support organizations |
| Weather Information | USS NEVERSAIL Operations Dept | USS NEVERSAIL Operations Dept, Shore-based Weather Support, Battle Group or Expeditionary Strike Group Weather | Higher authority, other Battle Group and Expeditionary Strike Group assets, ship's company, shore-based support organizations |
| Publications and Equipment Technical Manuals | USS NEVERSAIL Operations Dept | USS NEVERSAIL Administrative, Operations, Engineering, and Supply Depts | Ship's company, shore-based support organizations |
| Email messaging system | USS NEVERSAIL Operations Dept | USS NEVERSAIL Operations Dept | Ship's company |
| Chat | USS NEVERSAIL Operations Dept | USS NEVERSAIL Operations Dept | Ship's company watchstanders, specific Battle Group and Expeditionary Strike Group watchstanders |

Table 8 Assigned User Roles to Resources

c. Interface Requirements

Suppose the USS NEVERSAIL contracts a professional web-design organization whose specialty is usability and interface design. In addition, the U.S. Navy's Task Force Web provides templates, and USS NEVERSAIL personnel provided guidance to the contracted web design team to develop the vortal interfaces. Interfaces should be designed for interoperability. For example, interface design could use preprocessor dynamic web languages such as Microsoft's application server page (ASP), SUN Microsystem's java server page (JSP), or personal home pages (PHP).

d. Hardware Requirements

Hardware needs to be compatible and compliant with organizational information system requirements such as the IT21 standard. For the USS NEVERSAIL, availability requirements of three required resources are high. Table 9 displays the prioritized USS NEVERSAIL vortal content resources with their recommended availability. The required availability of these resources can be obtained through mechanisms such as clustering and external redundancy.

Chat functionality is of highest priority and requires a chat server. This functionality can be done with access to chat servers located onboard as well as externally. To support availability, the servers should be redundant, load-balanced, and clustered. This would allow for hardware that is scalable and accessible, and the chat server cluster would be seen as a single virtual server to users. Further, this hardware configuration would support graceful degradation of services if servers fail.

| Resource | Availability | Importance Ranking |
|--|---------------------|---------------------------|
| Equipment Casualty Reporting Information | Normal | 6 |
| Training Status Information | Normal | 7 |
| Fuel and Weapon Status Information | Normal | 5 |
| Troublecall Information | Normal | 8 |
| Parts Ordering | Normal | 4 |
| Weather Information | High | 3 |
| Publications and Equipment Technical Manuals | Normal | 9 |
| Email messaging system | High | 2 |
| Chat | High | 1 |

Table 9 Prioritized USS NEVERSAIL Vortal Resources With Associated Availability Requirements.

We recommend that each vortal server have two to four Intel Itanium Pentium (P4) multiprocessors with at least one Gigabyte of RAM and 160 GB of hard disk drive space. The Intel Itanium processor is designed to maximize support for parallel processing. Greater processing efficiencies may be achieved in software that takes advantage of this capability.

Using Raid technology, multiple hard-disk drives on each of the servers can be configured to support mirroring (redundancy of data), striping (increased speed of

data access), or a combination. In addition, multiple network-interface cards in each server would allow segregation of domains and further granularity of control.e

e. Software Selection

We suggest Plumtree portal software for the USS NEVERSAIL's vortal-management software for these reasons:

- Modular design. This enables flexibility and scalability within a particular vortal service, permitting additional vortal services with relative ease.
- Well established module technology. This ensures that other vendors have compatible software modules.
- Massive parallel portal engine technology to initiate job tasks and service.
- User access mode expandability (e.g. addition of wireless access).
- Portal server software compatibility with IT21 standards including the Windows NT operating system.
- Delivery of only HTML to the web browser. This makes portal output compatible with any legacy web browser, reduces required code complexity on the client side, and reduces network traffic.
- Connectivity to disparate resource and application server systems across the system-high classified network using common interoperable protocols (e.g. TCP/IP, HTTP, SOAP and LDAP) as well as provide flexibility for services growth.
- Interoperability with Java applications on systems with varied operating systems, enabling interoperability with legacy systems.

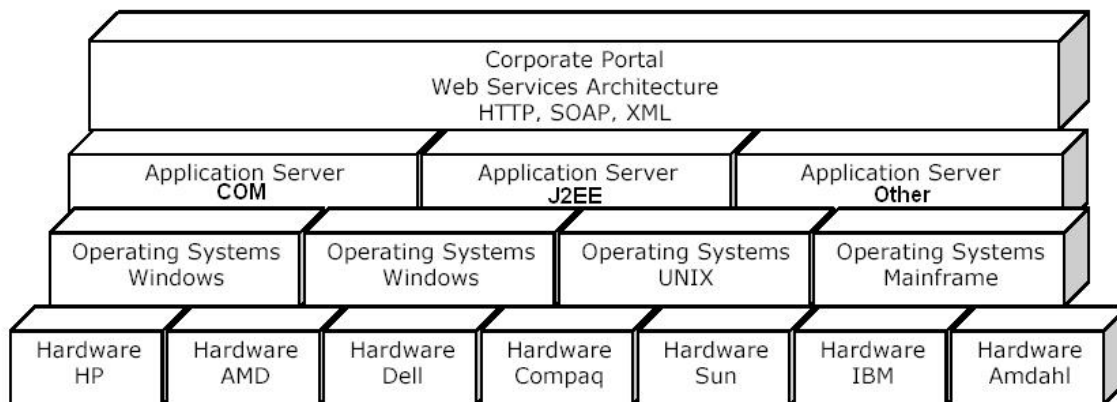


Figure 33. Web Service Provides Interoperability Across Application Servers (From Plumtree, 2002b)

Figure 33 describes the general interoperability capabilities of the Plumtree portal. As noted in the figure, several hardware manufacturers are supported.

5. Vortal Security Policy

The Plumtree product addresses the security policy in terms of roles, groupings, and content profiles. Table 10 correlates USS NEVERSAIL's user content levels with resources.

| Resource | Availability | Importance Ranking | User Content Level |
|--|---------------------|---------------------------|---------------------------|
| Equipment Casualty Reporting Information | Normal | 6 | 1 |
| Training Status Information | Normal | 7 | 1 |
| Fuel and Weapon Status Information | Normal | 5 | 1 |
| Troublecall Information | Normal | 8 | 2 |
| Parts Ordering | Normal | 4 | 2 |
| Weather Information | High | 3 | 1 |
| Publications and Equipment Technical Manuals | Normal | 9 | 2 |
| Email messaging system | High | 2 | 2 |
| Chat | High | 1 | 2 |

Table 10 Correlation of User Content Levels to Resources

Although the network hosting the USS NEVERSAIL vortal is a system-high classified network, content domains are organized based on content sensitivity (e.g. Classified, For Official Use Only, and Unclassified). To ensure that content remains accessible to the appropriate individuals, roles, groupings and content profiling is used. All information is treated as classified; however, user access to the content is discriminated based on need-to-know and clearance level. Need-to-know is determined by user group; clearance level by user role. The combination of a user's role and

assigned groups determine their associated content profile. Content profiles are used by the portal servers to build the content that users are entitled to view and authorized to use.

Availability of content is based on requirements. Following Table 10:

- Content is divided into three levels based on the target audience, giving two user levels and one content management level.
- Content availability requirements are identified based on applicable target audience needs.
- Rankings of resources indicate the priority of availability during reduced system capability as during maintenance and casualty response.

To support network security architecture, firewalls will be supplemented with internal security policies and measures. For example, during normal operation, the web server will run under the restricted rights and privileges of the process space “vortal user”, not under administrative privileges. This minimizes the risks of the “vortal user”.

Vortal servers will communicate to each other over HTTP(S). Search services will communicate over a specified TCP/IP port. Native API’s will be used by vortal servers to “communicate with internal resources such as file systems, domain controllers, or applications” (Plumtree, 2002c). Data transferred between browsers and servers or between servers will be encrypted using 128 bit SSL encryption. Data stored in databases or configuration files (persistent data) will use the vendor-specific security features as well as encryption such as the RSA 128 bit RC4 encryption algorithm to encrypt.

As per the vortal goal definition, access to the USS NEVERSAIL’s vortal will be through an intranet over the dedicated system-high classified network. Users can only gain access to the vortal is if they have access to the classified system-high network intranet where the vortal resides. But a single sign-on will be implemented for user access.

Figure 34 provides the basic logical diagram for the USS NEVERSAIL vortal using Plumtree technology. The Plumtree core management system (corporate portal software) provides the necessary mechanism to implement it via Web services for integration of search engines, applications, content, and users.

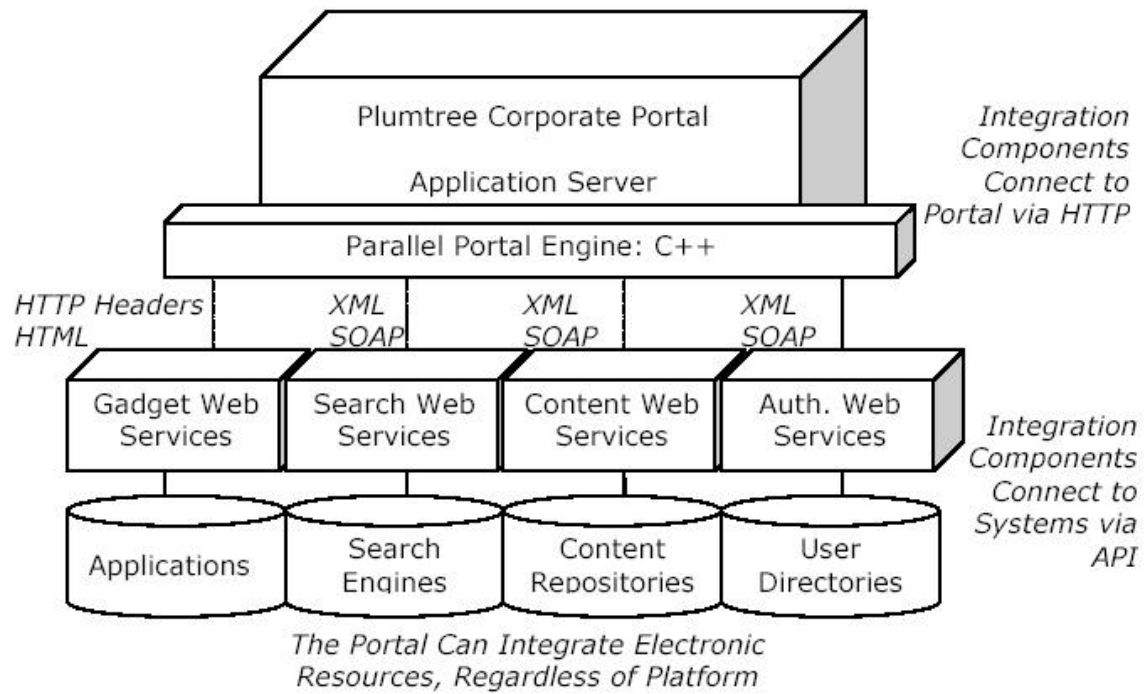


Figure 34. USS NEVERSAIL Logical Vortal Architecture (From Plumtree, 2002b)

D. ARCHITECTURE

Figure 35 displays the architecture developed for the USS NEVERSAIL vortal.

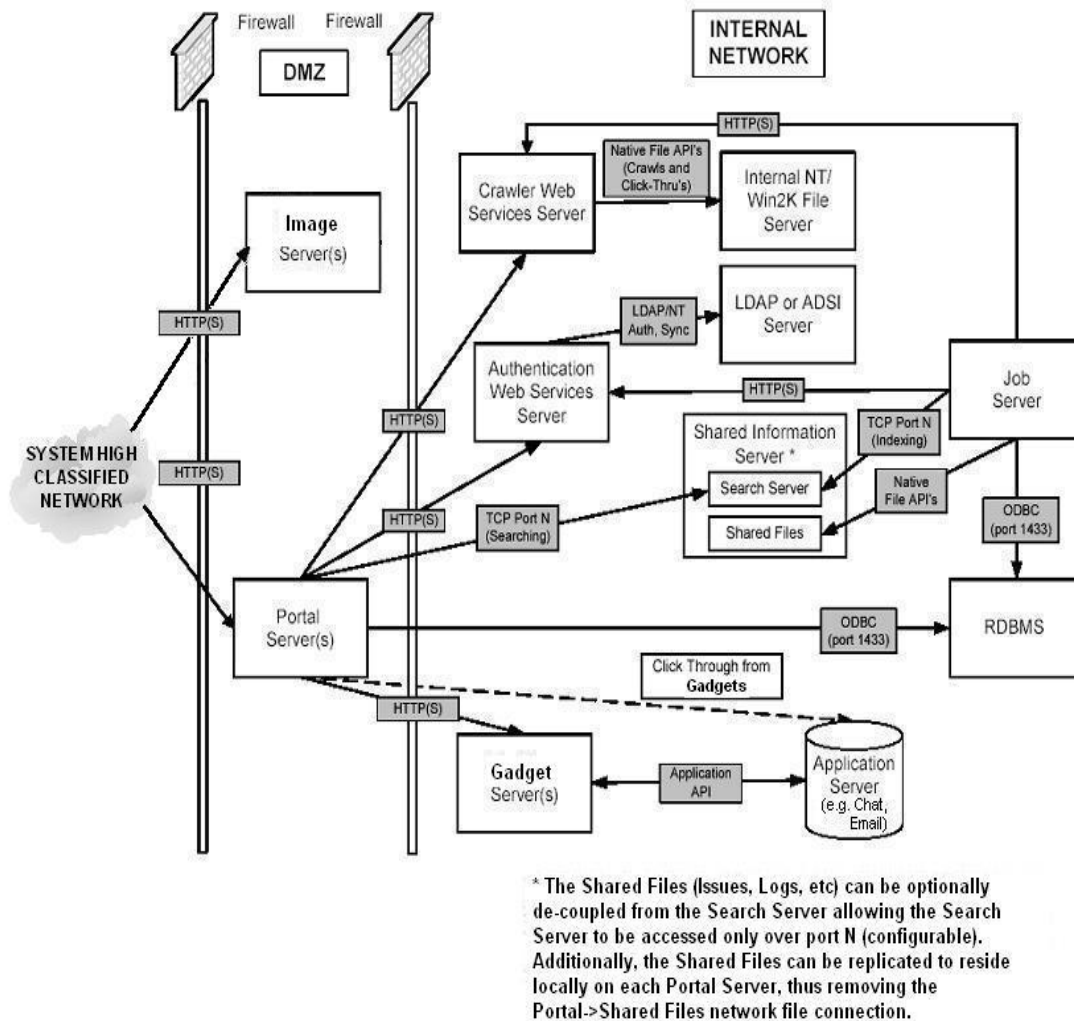


Figure 35. Proposed USS NEVERSAIL Vortal Architecture (Adapted From Plumtree, 2002c)

1. Architecture

Content domains should be created to segregate content based on user content levels noted in Table 10. For example, a user grouping known as “higher authority” may access reports generated from equipment casualty information, training information, and fuel and weapons information, but not the chat functionality.

Data base management systems and other relational databases should be relegated to their own content domain and should have limited direct access. All access to these databases should be through limited, pre-defined transactions and scripts that are

executed by a trusted agent such as a privileged internal server. Special data access requirements apply to users over the satellite channel.

Figure 36 illustrates the proposed security architecture. Content rings categorically group certain data and applications available to a specific target audience. The thick dark line which separates the content rings represents physical separation of services and content that is determined by physical security requirements. The terms *critical* and *non-critical* differentiate various types of data and applications, and could be determined by a system of categorization such as the classical Department of Defense information sensitivity categories like Top Secret, Secret, Confidential, Unclassified, and For Official Use Only.

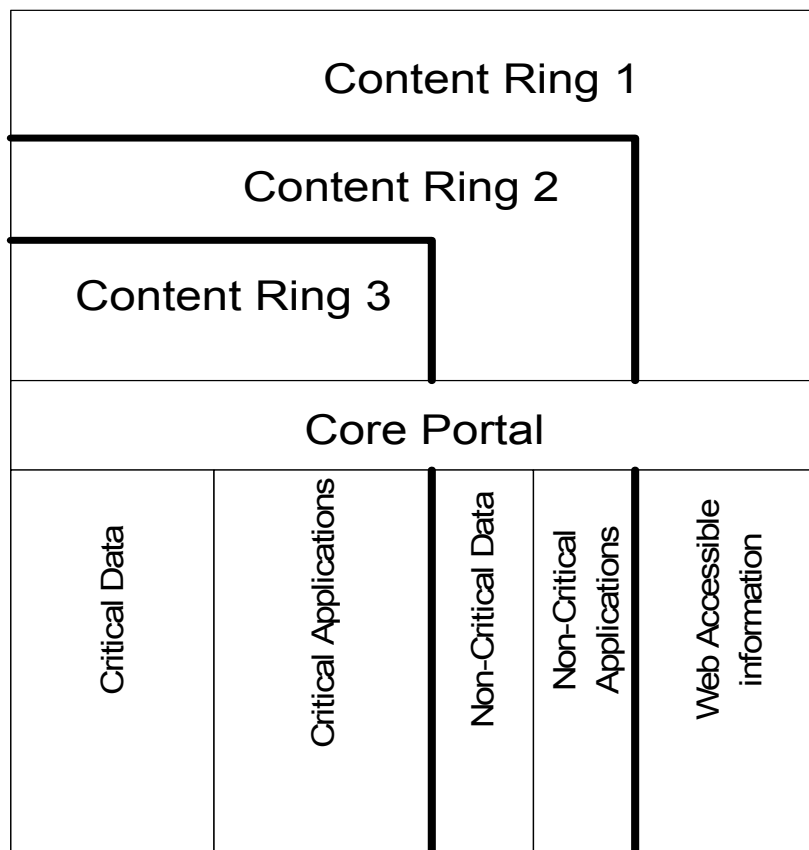


Figure 36. Proposed USS NEVERSAIL Vortal Security Architecture

Content ring three is devoted to content management. Therefore it is only visible to users that authenticate, can view this content as a part of the applicable entitlement user groupings, and fulfill the role of content managers or content maintainers. Content

ring two is devoted to internal USS NEVERSAIL personnel and is visible to those authenticated users who are part of the applicable entitlement groupings and fulfill the role of browsing user. Content ring one is devoted to the authenticated internal and external USS NEVERSAIL vortal audience, are a part of the applicable entitlement groupings, and are visible to users who fulfill the role of browsing users.

Security-policy critical-application points can be defined at the accesses to vortal databases as well as application (e.g. email, chat) as well as the portal servers that assemble the user entitled vortal content.

- The six main security services provided within the USS NEVERSAIL vortal should be authentication, confidentiality, integrity, access control, non-repudiation, and availability.
- Authentication: We suggest this service be performed through public-key infrastructure and Kerberos at initial session sign-on. This authentication method provides single sign-on ability.
- Confidentiality: We suggest this service be performed through the use of HTTP(S) and SSL encryption. In addition, databases and data repositories have vendor-specific security mechanisms and encryptions such as RSA 128 bit RC4 encryption. Further, the overall network is a system-high classified network which has bulk encryption provided by National Security Agency approved cryptographic equipment and key material.
- Integrity: We suggest this service be managed through user roles and entitlement groups. These mechanisms afford granularity of control and enable content managers the ability to assign to users all authorized actions they may perform on the data or applications to which they are entitled to view.
- Access control: We suggest this service be managed through content profiles associated with each vortal user. If the user is entitled to view or act on the particular data or application, then it will appear in their content profile.
- Non-repudiation: We suggest this service be managed in the transaction audits conducted within the confines of the organizational information systems security policies. All transactions are logged for audit.
- Availability: We suggest this service be provided only to the high-availability resources as determined by USS NEVERSAIL leadership. Servers that manage this content are configured, load-balanced, and clustered for traffic accommodation to ensure that resources are accessible and usable on demand by entitled authenticated users.

2. Design Assessment

Assessment scope definition is the first step of any assessment. For example, time latency in satellite connectivity to the USS NEVERSAIL at sea may present significant limitations with regards to software running on its three networks. Other USS NEVERSAIL issues that could be considered for assessment are software bandwidth and throughput requirements for the vortal, the impact of satellite channel bulk encryption on network performance within the constraints of the vortal, and the impact of satellite timing issues on authentication methods used in the vortal.

Metrics define exactly what is to be measured and the limits of the measurement. For example, 15 milliseconds of throughput latency may have a significant impact on some software communications within the network. A measure of performance could be this 15-millisecond latency threshold for network throughput.

E. PROTOTYPING

Prototyping the weather information and associated weather services functionality of the vortal would provide a small yet important project that has significant possibilities to yield success. Available weather sites and weather related resources on the network could be made available through the vortal. In addition, other possible prototypes could deal with email, part ordering, chat, or trouble calls, and could provide building blocks toward successful implementation of the vortal.

F. DEVELOP TRAINING, MAINTENANCE & ADMINISTRATION PLANS

The development of various plans to accommodate training, maintenance and administration is essential to the success of a vortal effort. Quite often during the development of these plans, problem areas arise that present challenges to overcome for the assembled vortal team. In the case of the USS NEVERSAIL vortal, certain aspects related to training, maintenance, and administration will present difficulties during development and implementation efforts.

It is important to note that training is can significantly help in the development and implementation of a vortal solution; however, it is not a panacea. Change associated with the incorporation of new technologies can be difficult for organization members to accept. It seems that the longer a person has been with an organization, the more difficult

it is to change their behaviors. Training may provide a mechanism to accommodate some of this change but it does not always solve the problems. For example, it does no good for the head of the organization to declare that an organization will go paperless, train all members of an organization on the new paperless system, and then keep the infrastructure for the old paper-producing system intact. Familiarity with the old system will inevitably allow some individuals the avenue to overcome the new system. Training cannot overcome deeply rooted organizational behaviors if the infrastructure supporting those behaviors remains intact.

Training cannot overcome challenges presented by a system that is not available due to a lack of maintenance or a maintenance schedule that does not support the goals of the vortal. The thorough training of back-office maintainers is vital to the health of the network and vortal. With that said, unless the maintainers apply that training to maintain the network as well as vortal hardware and software the health of the system will dictate availability to the user. A less than thoroughly trained maintainer will leave a trouble call customer with a less than thoroughly satisfying experience. Moreover, the hardware and software may be damaged if the maintainers do not know what they are doing.

Maintenance presents another particular challenge to the development and implementation of a vortal. If a vortal is designed to be easily maintained, then challenges may be minimized. However, not all vortal designs can be deployed. It may be that an existing network dictates some of the vortal design. Maintenance may become significantly more difficult if the vortal architecture does not align with the network architecture to which it resides. Conflicts and incompatibilities may arise which prevent the smooth delivery of information and services. Further, with the addition of new resources to the vortal, maintenance issues may become more complex. For example, if a particular resource has a hard-coded synchronization timeframe that does not coincide with other network required synchronizations, traffic over the network may be negatively impacted for a larger percentage of the computing day. In addition, this synchronization traffic could increase the network traffic greater than acceptable limits. Consequently, maintenance is directly tied to the developed network and vortal architectures.

Administration of the vortal is impacted by the architecture of the vortal, maintenance performed on the system, and the training provided to administrators. As mentioned before, if the vortal architecture does not support efficiency in the administration of the vortal then complexities may develop that will create significant problems later. For example, if the process for the addition of vortal users uses an automated script and the script does not perform properly due to a change in vortal architecture, then user content profiles may be created in a manner that prevents the subsequent migration of these profiles to an upgraded system. If this problem affects 20 users it is not significant. However, if there are 56,000 users then this could require a considerable amount of man-hours to solve.

Administration of a system that is poorly maintained may take a considerable amount of time and other resources. For example, if there is not procedure for deletion of invalid users then the accumulation of unused profiles may exceed system resources. This depletion of system resources may not be significant when there are an excessive amount of resources available; however, when resources are low, user administration may become a considerable undertaking. In addition, the performance of the system is related to maintenance. Decreased performance is usually associated with a system that is poorly maintained. Administration of content and other resources beyond users may become almost impossible if the system is not available due to poor maintenance.

Training for administration of the vortal is very important. A system may function properly and the architecture may be well developed, however, if the administrators are poorly trained, they may break the system and consequently impact vortal availability.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. COMPUTER SUPPORT FOR DOCTRINE AND POLICY

In the dynamic hybrid information environment of the 21st century, doctrine and policy can be both large and complex. They may change frequently and may not be complete; they may have gaps (Michael, J. 2001). Is it necessary to check for holes in policy and doctrine or the impact of a particular change within them? As discussed before, they do provide direct guidance in the formulation of the organization's mission as well as organization policies. Changes to these may not be easily discernible and may have an impact on overall organization requirements as well as security requirements. Unless these changes are analyzed for impact (i.e. what they do and do not provide) then an organization's information system and its associated security requirements may include unnecessary things that can translate into embedded vulnerabilities.

It is possible to create automated policy-governing tools which reason and maintain policy at high levels of performance. In a network-centric warfare environment, the Navy can gain a competitive advantage through being able to quickly refine new or changing policy and doctrine into requirements and other system artifacts. Automated policy-governing tools have important implications in the formulation and maintenance of vortal policies including its security policy.

Vortal security policies are developed from vortal requirements and the organization's information-system security policy, and its formulation is impacted by higher-level policies and doctrine as well as organization policies and mission. The delayed propagation of higher-level document changes can negatively impact security of a future or current vortal solution. This suggests that these automated policy-governing tools could appropriately respond to dynamic changes in the hybrid information environment or higher-level documents. Consequently, they could be used to assist with the rapid development and maintenance of vortal security requirements. This could significantly reduce or eliminate embedded vulnerabilities in a current or future system.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSIONS

The required quantity and delivery speeds of vital data, information and services necessary to adequately support the Navy greatly exceeds what is currently provided. To support the Chief of Naval Operation's vision of "Sea Power 21" and its component Navy initiatives, mechanisms such as vortals that allow for the timely collaboration, integration, aggregation and consolidation of vital data, information, and services can be critically important. Vortals, for which we can apply security at the concept stage of the development process, can provide the required framework to successfully support current and future Navy initiatives. Consequently, the study of security architecture, doctrine, policy and implementation for vortals is needed to develop innovative and transformational ways to securely implement these essential mechanisms.

Vortals provide, through a single user interface, specific narrow-scoped data, information, and services to a specific target audience. The secure vortal architecture presented provides a framework for the development of concepts for successful implementation of secure vortals as well as a starting point for additional efforts. Further, the provided secure vortal development diagram and the adapted MODELS Information Architecture provide a context as well as mechanisms for the successful implementation of secure vortal architecture.

A. FUTURE WORK

Vortal security research presents many opportunities to make advances in a field that holds much promise for both the Department of the Navy as well as the private sector.

- Effective and efficient policy and doctrine within the context of vortal security needs further study. Doctrine provides the context within which policy may be applied; policy codifies the accepted practices for vortal implementers. Doctrine and policy, taken together, enable the standardization of successful implementation strategies across a set of specific vortal solutions. In addition, doctrine also provides the context to apply policy within a specific set of limitations such as those outlined by the vortal security architecture and the secure vortal development diagram of this thesis. Further, automated tools which formulate requirements

from doctrine and policy may enable the rapid vortal solution development necessary to fulfill critical just-in-time user requirements.

- The key to the effectiveness of a secure vortal implementation lies in prototyping and testing. Prototyping and testing provides additional data that can be used for implementation refinement purposes. Additionally, prototyping and testing can be used to prove the effectiveness of applied theory. More research needs to be applied to these topics.

The methodology presented within this document could prove valuable for other types of portals. For this premise to be proven correct, scalability prototyping and testing must be conducted. Prototyping and testing may provide refinement and proposed-change inputs to existing methodology which may be necessary for other portals.

APPENDIX A. LIST OF ACRONYMS

| | |
|------------|--|
| ABUSE Case | A particular type of UML Use Case that focuses on the behavior that may be harmful to a computing system. |
| ACI | Autonomy Content Infrastructure; proprietary plug-in technology for communications between enterprise applications |
| ACL | Access Control List |
| API | Application Programming Interface |
| ASP | Application Server Page |
| B2E | Business to Employees |
| CA | Certificate Authority; the trusted service authority in public key infrastructure |
| CGI | Common Gateway Interface |
| CMS | Core Management System |
| COI | Critical Operational Issues |
| CRM | Customer Relationship Management |
| DBMS | Database Management System |
| DMZ | Demilitarized Zone; a common name for untrusted perimeter networks |
| DOC | Microsoft Word Document Format |
| DoD | Department of Defense |
| DITSCAP | Department of Defense Information Technology Security Certification and Accreditation Process |
| DRE | Dynamic Reasoning Engine; a proprietary technology from Autonomy |
| EJB | Enterprise Java Bean; a proprietary technology from SUN Microsystems |
| ERP | Enterprise Resource Planning |
| GB | Gigabyte |
| GUI | Graphical User Interface |
| HCI | Human-Computer Interaction |
| HTML | Hypertext Markup Language |
| HTTP | Hypertext Transport Protocol |
| HTTP(S) | Hypertext Transport Protocol, Secure |

| | |
|--------|--|
| IP | Internet Protocol |
| IT21 | Information Technology 21 standard |
| JSP | Java Server Pages |
| J2EE | Java 2 Platform, Enterprise Edition |
| JTA | Joint Technical Architecture |
| KDC | Key Distribution Center; trusted authority in Kerberos authentication |
| LDAP | Light-weight Directory Access Protocol |
| MIA | MODEL Information Architecture |
| NNTP | Network News Transfer Protocol |
| NTLM | Windows NT Logon Management |
| ODBC | Open Database Connectivity |
| OID | Oracle Internet Directory |
| ORIS | Organizational Information System |
| PHP | Personal Home Page; an open source pre-processor language similar to ASP or JSP |
| PKI | Public Key Infrastructure |
| RAM | Random Access Memory |
| RDBMS | Relational Database Management System |
| RTF | Rich Text Format |
| SGML | Standard Generalized Markup Language |
| SOAP | Simple Object Access Protocol |
| SQL | Structured Query Language |
| SSL | Secure Socket Layer |
| SSO | Single Sign-On |
| ST | Service Ticket; used in Kerberos authentication to enable the session use of a particular service |
| TCP | Transport Control Protocol |
| TF WEB | Task Force Web |
| TGT | Ticket Granting Ticket; used in Kerberos authentication to enable the user to be able to request an ST for use of particular service |
| UDDI | Universal Description, Discovery and Integration |
| UML | Universal Modeling Language |
| URL | Universal Resource Locator |

| | |
|----------|--|
| Use Case | A UML mechanism used to describe a complete transaction within a system and its actors |
| Vortal | Vertical Industry Portal |
| VPD | Virtual Private Database |
| WebDAV | Web-based Distributed Authoring and Versioning |
| WSDL | Web Service Description Language |
| X.509 | Standard for defining digital certificates which enable PKI based authentication. |
| XML | Extensible Markup Language |

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. HUMAN-COMPUTER INTERACTION AND USABILITY RESOURCES

- Ask Tog: First Principles. <http://www.asktog.com/basics/firstPrinciples.html>
- Bad Human Factors Designs. <http://www.baddesigns.com/>
- Bastien, C., Scapin, D. & Leulier, C. [Looking for Usability Problems with the Ergonomic Criteria and with the ISO 9241-10 Dialogue Principles.](#)
- Bevan, N. [Human-Computer Interaction Standards.](#)
- Bevan, N. [Usability Issues in Website Design.](#)
- Bevan, N., Kirakowski, J. & Maissel, J. [What is Usability?](#)
- Bevan, N. & Macleod, M. [Usability Measurement in Context.](#)
- Flanders, V. 1998. *Web Pages That Suck: Learn Good Design by Looking at Bad Design.* <http://www.webpagesthatsuck.com/> New York, NY: Sybex.
- HCI Bibliography. <http://www.hcibib.org/>
- HCI Resources on the Net. <http://www.ida.liu.se/labs/aslab/groups/um/hci/>
- Homl, J. [Usability Methods Toolbox.](#)
- Instone, K. How to Test Usability .
- Interface Hall of Shame. <http://www.iarchitect.com/shame.htm>
- Kirakowski, J. [The Use of Questionnaire Methods for Usability Assessment.](#)
- Kirakowski, J. & Cierlik, B. [Measuring the Usability of Websites.](#)
- Kirakowski, J., Claridge, N. & Whitehand, R. [Human Centered Measures of Success in Website Design.](#)
- Lynch, P. and Horton, S. 1999. *Web Style Guide: Basic Design Principles for Creating Web Sites.* <http://info.med.yale.edu/caim/manual/contents.html>. Yale University Press.
- Macleod, M. [Usability: Practical Methods for Testing and Improvement.](#)
- Macleod, M. [Usability in Context: Improving Quality of Use.](#)
- Macleod, M., Bowden, R. & Bevan, N. [The Music Performance Measurement Method.](#)

Macleod, M. & Rengger, R. [The Development of DRUM: A Software Tool for Video-Assisted Usability Evaluation.](#)

Melchior, E. et al. [Usability Study.](#)

Morkes, J. & Nielsen, J. [Concise, Scannable, and Objective.](#)

Nielsen, J. [Characteristics of Usability Problems Found by Heuristic Evaluation.](#)

Nielsen, J. [Heuristic Evaluation.](#)

Nielsen, J. [Ten Usability Heuristics.](#)

Nielsen, J. [Cost of User Testing a Website.](#)

NIST. [Common Industry Format for Usability Test Reports.](#)

Usability First. <http://www.usabilityfirst.com/>

APPENDIX C. DECISION MAKING RESOURCES

Brassard, M. 1989. *The Memory Jogger Plus+*. Methuen, MA: GOAL/QPC.

Forsberg, K. 1996. *Visualizing Project Management*. New York, NY: John Wiley & Sons, Inc.

Mizuno, S. 1988. *Management for Quality Improvement: The Seven New QC Tools*. Cambridge, MA: Productivity Press.

Scholtes, P. 1988. *The Team Handbook*. Madison, WI: Joiner Associates.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

- Autonomy. 2002. *Autonomy Portal Infrastructure*. [online] December 2002.
[<http://www.autonomy.com/Extranet/Marketing/Autonomy%20White%20Papers/Autonomy%20Portal%20Infrastructure%20WP.pdf>]
- Autonomy. 2002a. *Autonomy Security*. [online] December 2002.
[<http://www.autonomy.com/Extranet/Marketing/Autonomy%20White%20Papers/Autonomy%20Security%20WP.pdf>]
- Brewer, E. 2001. *Lessons from Giant-Scale Services*. IEEE Internet Computing, Vol. 5 Issue 4, pp. 46–55.
- Brio. 2001. *Brio Portal 7.0*. [online] December 2002.
[http://www.brio.com/pdfs/Brio_DataPortal_10-10.pdf]
- Castano, S. 1995. *Database Security*. Harlow, England: Addison-Wesley.
- Cohen, F. 1998. *A Note on the Role of Deception in Information Protection*. Deception for Protection. [online] December 2002. [<http://all.net/journal/deception/deception.html>]
- Collins, D. 1999. *Data Warehouses, Enterprise Information Portal, and the SmartMart Meta Directory*. Information Builders Systems Journal, 12(2), pp. 53-61.
- Collins, H. 2001. *Corporate Portals: Revolutionizing Information Access to Increase Productivity and Drive the Bottom Line*. New York, NY: American Management Association.
- Department of Defense. 2001. *Joint Technical Architecture, Volume 4* (April) [online], December 2002. [http://www-jta.itsi.disa.mil/jta/jtav4_dnld.html]
- Dias, C. 2001. *Corporate Portals: A Literature Review of a New Concept in Information Management*. International Journal of Information Management 21, pp. 269-287.
- Gardner, T. with Miller, P. and Russell, R. 1999. *The MIA Logical Architecture*. UKOLN Ver. 0.3. (September) [online] December 2002.
[<http://www.ukoln.ac.uk/dlis/models/requirements/arch/>]
- Gardner, T. with Miller, P. and Russell, R. 1999a. *MIA Functional Model*. UKOLN Ver 0.3, (September) [online], December 2002.
[<http://www.ukoln.ac.uk/dlis/models/requirements/func/>]
- Gasser, M. 1988. *Building a Secure Computer System*. New York, NY: Van Nostrand Reinhold.
- Gupta, M. 2001. *The Experimental Analysis of Information Security Management Issues for Online Financial Services*. Indianapolis, IN: Purdue University Press.

Hewett, T. 1992. *ACM Sigchi Curricula for Human Computer Interaction*. New York, NY: Association for Computing Machinery.

International Standards Office. 1998. Security. Section 08. Second Edition. ISO/IEC 2382.

Kabay, M. 1996. *The NCSA Guide to Enterprise Security: Protecting Information Assets*. New York, NY: McGraw-Hill.

Kargl, F. 2001. *Protecting Web Servers from Distributed Denial of Service Attacks*. New York, NY: ACM Press.

Larman, C. 1998. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design*. Upper Saddle River, NJ: Prentice Hall.

Lowman, T. and Mosier, D. 1997. Applying the DoD Goal Security Architecture As a Methodology for the Development of System and Enterprise Security Architecture. In *Proc. 13th Computer Security Applications Conference*, San Diego, CA, December 1997.

Machiraju, V. 1996. *A Survey on Research in Graphical User Interfaces*. Department of Computer Science University of Utah, Salt Lake City, UT (August) [online], December 2002. [<http://citeseer.nj.nec.com/machiraju96survey.html>]

McDermott, J and Fox, C. 1999. Using Abuse Case Models for Security Requirements Analysis. In *Proc. 15th Annual Computer Security Applications Conference, ASCAC*, Harrisonburg, VA, December 1999.

McGraw, G. 1998. *Testing for Security During Development: Why We Should Scrap Penetrate-and-Patch*. IEEE Aerospace and Electronics Systems Magazine, Vol 13, Issue 4, pp. 13–15 [online], December 2002. [<http://ieeexplore.ieee.org/ie13/62/14634/00666831.pdf?isNumber=14634&prod=JNL&arNumber=00666831>].

McGraw, G. and Viega, J. 2002. *Building Secure Software: How to Avoid Security Problems the Right Way*. New York, NY: Addison-Wesley.

McGraw, G. 2002. *Managing Software Security Risks*. Computer, Vol. 35, Issue 4, pp. 99–101. (April) [online], December 2002. [<http://ieeexplore.ieee.org/ie15/2/21439/00993782.pdf?isNumber=21439&prod=JNL&arNumber=00993782>].

Michael, J. 2001. Natural-Language Processing Support for Developing Policy-Governed Software Systems. In *Proc. 39th International Conference on Technology for Object-Oriented Languages and Systems*, IEEE Computer Society Press, Santa Barbara, CA. July 30 through August 2, 2001.

Microsoft, 2002. *SharePoint Portal Server Product Overview*. (June) [online], December 2002. [<http://www.microsoft.com/sharepoint/evaluation/overview/>]

Nielsen, J. 2000. *Designing Web Usability: The Practice of Simplicity*. Indianapolis, IN: New Riders Publishing.

Oracle, 2002. *ORACLE9iAS Portal Release 2 Tech Overview* [online], December 2002. [http://portalstudio.oracle.com/pls/ops/docs/FOLDER/COMMUNITY/OTN_CONTENT/MAINPAGE/KEYFEATURES_BENEFITS/PORTAL_V2_TWP_FINAL.PDF]

Oracle, 2002a. *ORACLE9i Application Server Portal* [online], December 2002. [http://www.oracle.com/ip/deploy/ias/portal/portal_overview.pdf]

Pfleeger, C. 1997. *Security in Computing*, 2nd Edition. Upper Saddle River, NJ: Prentice Hall.

Powell, A. 2000. *DNER Portal Architecture*. RDNC Revision 1.7, Draft (27 March) [online], December 2002 [<http://www.rdn.ac.uk/publications/mia/>]

Plumtree, 2000. *Corporate Portal Security: Corporate Portal Security Requirements and the Plumtree Corporate Portal* [online], December 2002. [http://img.plumtree.com/pdf/Security_White_Paper.pdf]

Plumtree, 2002. *Plumtree Corporate Portal Detailed Architecture* [online], December 2002. [<http://www.plumtree.com/products/platform/>]

Plumtree, 2002a. *The Web Services Architecture of a Corporate Portal: Web Services, Application Servers and Corporate Portals* (May) [online], December 2002. [http://img.plumtree.com/pdf/Web_Services_Architecture_White_Paper_May02.pdf]

Plumtree, 2002b. *The Plumtree Corporate Portal's Standards Support: Using Web Services Standards to Overcome Traditional Divisions* [online], December 2002. [http://img.plumtree.com/pdf/Plumtree_Standards_White_Paper.pdf]

Plumtree, 2002c. *Deploying the Plumtree Corporate Portal 4.5WS to the Extranet*. (August) [online], December 2002. [http://img.plumtree.com/pdf/Q3_2002_WP_Deploying_Corporate_Portal_Extranet.pdf]

Reynolds, H. & Koulopoulos T. 1999. *Enterprise knowledge has a face*. Intelligent Enterprise, 2(5), 29-34, (March) [online], December 2002. [http://www.intelligententerprise.com/db_area/archives/1999/993003/feat1.shtml].

Russell, R. with Gardner, T. and Miller, P. 1999. *MIA Requirements Analysis Study: Hybrid Information Environments - Overview and Requirements*. UKOLN. Ver 0.1 Draft, (22 July) [online], December 2002. [<http://www.ukoln.ac.uk/dlis/models/requirements/overview/>]

- Schneier, B. 1998. *Cryptographic Design Vulnerabilities*. Computer, 31(9), 29-33 (September) [online], December 2002. [<http://www.counterpane.com/design-vulnerabilities.pdf>].
- SearchCIO.com. 2002. "Vortal" [online], December 2002. [http://searchCIO.techtarget.com/sDefinition/0,,sid19_gci213601_00.html]
- Silberschatz, A. and Galvin, P. 2001. *Operating System Concepts, Sixth Edition*. New York, NY: John Wiley & Sons.
- Stallings, W. 2000a. *Network Security Essentials: Applications and Standards*. Upper Saddle River, NJ: Prentice Hall.
- Task Force Web 2002. *Mission* [online], October 2002. [[https://ucso2.hq.navy.mil/n09w/webbas01.nsf/\(vwWebPage\)/WebBase.htm?OpenDocument&Set=1](https://ucso2.hq.navy.mil/n09w/webbas01.nsf/(vwWebPage)/WebBase.htm?OpenDocument&Set=1)]
- Webopedia.com. 2002. "Web Portal" [online], December 2002. [http://www.webopedia.com/TERM/W/Web_portal.html]
- White, C. 1999. *Using Information Portals in the Enterprise*. DM Review (April). [online], December 2002. [<http://www.dmreview.com/master.cfm?NavID=55&EdID=61>]
- White, C. 2001. *Safeguarding the Corporate Portal: A Review of Portal Security*. Database Associates Whitepaper (January) [online], December 2002. [<http://www.dbaint.com/pdf/Viador%20Security%20Paper%20V1.2%20Screen.pdf>]
- White, C. 2002. *Critical Window*. Intelligent Enterprise (February) [online], December 2002. [http://www.intelligententerprise.com/020221/504feat1_1.shtml]

BIBLIOGRAPHY

Alberts, D. 1996. *Defensive Information Warfare*. Washington, D.C.: National Defense University: Institute for National Strategic Studies.

Brewer, E. 2001. *When Everything is Searchable*. New York, NY: Association for Computing Machinery.

Department of Defense. *Department of Defense Information Technology Security Certification and Accreditation Process (DITSCAP)*, DODINST 5200.40 (series)

Denning, D. 1999. *Information Warfare and Security*. Reading, MA: Addison-Wesley.

Duray, D. and Vering, M. 2001. *The E-Business Workplace: Discovering the Power of Enterprise Portals*. New York, NY: John Wiley & Sons.

Finkelstein, C. and Aiken, P. H. 2000. *Building Corporate Portals with XML*. New York, NY: McGraw-Hill.

Katz, R. N. 2002. *Web Portals and Higher Education: Technologies to Make IT Personal*. San Francisco, CA: John Wiley & Sons.

Landauer, T. 1999. *The Trouble with Computers: Usefulness, Usability, and Productivity*. Cambridge, MA: MIT Press.

Stallings, W. 2000b. *Data and Computer Communications*. Upper Saddle River, NJ: Prentice Hall.

Walker, A. 2000. *Knowledge Portal Support to the Naval Postgraduate School's Advanced Distributed Learning Program for the Information Systems and Operations Curriculum*. Monterey, CA: DTIC.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Ft. Belvoir, Virginia
2. Dudley Knox Library
Naval Postgraduate School
Monterey, California
3. IP Center of Excellence
Naval Postgraduate School
Monterey, California
4. Dr. Neil Rowe
Naval Postgraduate School
Monterey, California
5. Dr. Xavier Maruyama
Naval Postgraduate School
Monterey, California
6. LCDR Peter Wu
Commander, Third Fleet
San Diego, California
7. CAPT James Bird
Commander, Third Fleet
San Diego, California
8. CDR Chuck Gompf
Commander, Third Fleet
San Diego, California